# FINITE ELEMENT SOLUTION FOR AXISYMMETRIC TRANSIENT THERMAL STRESSES

Manouchehr Bakhshandehpour

NAVAL POSTGRADUATE SCHOOL
Monterey, California

# THESIS

FINITE ELEMENT SOLUTION FOR
AXISYMMETRIC TRANSIENT THERMAL STRESSES

by

Manouchehr Bakhshandehpour

Thesis Advisor: R. E. Newton

June 1972

# Finite Element Solution for Axisymmetric Transient Thermal Stresses

by

Manouchehr Bakhshandehpour
Lieutenant, Imperial Iranian Navy
B.S., Italian Naval Academy, 1960

Submitted in partial fulfillment of the
requirements for the degrees of

MECHANICAL ENGINEER

and

MASTER OF SCIENCE IN MECHANICAL ENGINEERING

from the

NAVAL POSTGRADUATE SCHOOL
June 1972

## ABSTRACT

A finite element formulation for solving axisymmetric transient heat conduction and thermal stress problems is developed in this thesis. The governing equations of uncoupled, linear, isotropic thermoelasticity are discretized using quadratic isoparametric elements. A FORTRAN IV program, using double precision arithmetic, is presented. Compact storage techniques for banded symmetric matrices are used.

Comparisons between exact and computer solutions demonstrate close agreement for a number of test problems. Detailed instructions for using the program are included.

## TABLE OF CONTENTS

## LIST OF FIGURES

## LIST OF TABLES

## LIST OF SYMBOLS

Note: A single underline is used to denote a column vector and a double underline denotes a rectangular matrix. The symbols used in computer program are described in the beginning of Appendix B.

| | |
|---|---|
| $a$ | Entrance fluid cross-sectional area |
| $\underline{\underline{A}}$ | Linear combination of $\underline{\underline{C}}$ and $\underline{\underline{Y}}$ matrices |
| $\underline{b}$ | A constant vector |
| $\underline{\underline{B}}$ | Standard rectangular strain-displacement matrix |
| $\underline{\underline{C}}$ | Thermal capacitance matrix |
| $c$ | Specific heat |
| $\underline{\underline{D}}$ | Standard elasticity matrix |
| $E$ | Young's modulus of elasticity |
| $e$ | Superscript designating element contribution |
| $\underline{F}$ | Load vector |
| $\underline{F}_T^e$ | Element thermal load vector |
| $\underline{F}_p^e$ | Element pressure load vector |
| $\underline{F}_c^e$ | Element centrifugal load vector |
| $\underline{\underline{G}}$ | Linear combination of $\underline{\underline{C}}$ and $\underline{\underline{Y}}$ matrices |
| $h$ | Surface heat transfer coefficient |
| $\underline{\underline{I}}$ | Identity matrix |
| $\underline{\underline{J}}$ | Jacobian coordinate transformation matrix |
| $\underline{\underline{K}}$ | System stiffness matrix |
| $\underline{\underline{K}}^e$ | Element stiffness matrix |
| $k$ | Thermal conductivity |
| $L$ | Thickness of slab |

| | |
|---|---|
| $\ell$ | Arc length along the side of quadrilateral |
| $N_i$ | Shape function |
| m | Total number of nodes |
| n | Outward normal or number of nodes per element |
| P | Pressure |
| R | Radial coordinate |
| S | Surface area |
| $\underline{T}$ | Nodal temperature vector |
| T | Temperature or, when used as a superscript, transpose of a matrix |
| $T_{avg}$ | Average temperature |
| $T_f$ | Fluid temperature (used in one-dimensional example) |
| $\underline{U}$ | A vector defined as $\langle 1\ 1\ 1\ 0\rangle^T$ |
| u | Radial displacement |
| $\underline{v}$ | Right-hand side vector in conduction equation |
| V | Volume |
| W | Work done by loads |
| $\underline{\underline{W}}$ | Modal matrix |
| w | Axial displacement |
| $\underline{w}$ | Eigenvector |
| $\underline{\underline{Y}},\underline{\underline{Y}}^+$ | Thermal admittance matrix |
| y | Dependent variable |
| $y^*_{ij}$ | Element ij of the matrix $\underline{\underline{Y}}^*$ |
| [ ] | Matrix representation |
| < > | Row vector |
| $\overline{\nabla}$ | Gradient operator |
| $\alpha$ | Coefficient of thermal expansion |
| $\underline{\beta}$ | Constant coefficient vector |

| | |
|---|---|
| $\underline{\delta}$ | Nodal displacement vector |
| $\underline{\delta}^e$ | Element displacement vector |
| $\underline{\varepsilon}$ | Strain vector |
| $\mu$ | Eigenvalue of one-dimensional transient temperature solution |
| $\underline{\varepsilon}_o$ | Thermal strain vector |
| $\underline{\varepsilon}^e$ | Element strain vector |
| $\eta$ | Local element coordinate |
| $\nu$ | Poisson's ratio |
| $\lambda$ | Eigenvalue |
| $\rho$ | Material density |
| $\underline{\sigma}$ | Stress vector |
| $\xi$ | Local element coordinate |
| $\tau$ | Time |
| $\Delta\tau$ | Step size of numerical time integration |
| $\tau_{RZ}$ | Shearing stress component |
| $\theta$ | Fluid temperature, or angle |
| $\underline{\underline{\Lambda}}$ | Spectral matrix |
| $\Omega$ | Speed of rotation |

## ACKNOWLEDGEMENTS

The author would like to express his gratitude to the Imperial Iranian Navy for having made it possible for him to undertake the course of graduate study leading to the present text.

To Dr. Robert E. Newton, Professor of Mechanical Engineering, the author wishes to express deep appreciation for his inspiring advice and guidance. Without his diligence and patience this study could not have been accomplished in its present form. The author is obliged to Dr. Gilles Cantin, Professor of Mechanical Engineering, for his generous advice during the course of study and computer programming. Thanks are also due to the personnel at the computer center of Naval Postgraduate School.

Finally the author should thank his wife for her patience and understanding throughout his work at this institution.

# I. INTRODUCTION

Thermal stresses have become increasingly important in engineering practice during recent years. In power generation higher cycle temperatures and use of nuclear fission are largely responsible for this trend. This thesis describes a computer program for finding temperatures and stresses in bodies having axisymmetric geometry and loading. The governing equations are those of isotropic, uncoupled, quasi-static, linear thermoelasticity. They are discretized by using the finite element method. A FORTRAN IV computer program using double-precision arithmetic has been written to solve problems of the following kinds.

## A. TEMPERATURE PROBLEMS

The transient temperature vector, evaluated at the nodal points, may be obtained for an axisymmetric body with the combination of insulated, convection, or constant temperature boundary conditions. For the convection thermal boundary condition, however, we may have fluid flowing with entry temperature prescribed as a linear function of time (RAMP). The program can handle up to 15 different ramps, each having a different flow velocity, and with discontinuities between successive ramps.

## B. STRESS PROBLEMS

The program will generate load vectors for pressure loading, centrifugal loading, and axial force. Provision is

made for direct input of one additional load vector. Stresses may be found for any combination of these loadings.

## C. THERMAL STRESS PROBLEMS

Thermal stresses may be found for as many as 20 different temperature vectors which may be output of the temperature problem or direct input. In short, in this part any combination of the temperature and stress problems may be used.

# II. FINITE ELEMENT FORMULATION OF HEAT CONDUCTION IN AXISYMMETRIC BODIES

## A. METHOD OF FORMULATION

For bodies of revolution under axisymmetric loading the mathematical problems presented are two-dimensional. The governing equation for non-steady heat conduction is

$$\overline{\nabla}\cdot k\overline{\nabla}T = \rho c\dot{T}, \tag{1}$$

where k is the thermal conductivity, $\rho$ the density, c the specific heat, T the temperature and $\overline{\nabla}$ the gradient operator. The superior dot denotes a time derivative.

Applying Galerkin's principle [1] gives

$$\int_V N_i\overline{\nabla}\cdot k\overline{\nabla}T \, dV = \int_V \rho c \, N_i \, \dot{T} \, dV, \tag{2}$$

where the integral is over the volume V of the conducting body and $N_i$ is a "shape function" used in representing the temperature distribution. If S is the surface of the body and n the outward normal to surface, then using Gauss' theorem we can write

$$\int_V \overline{\nabla}\cdot(N_i \; k\overline{\nabla}T) \, dV = \int_S N_i \, k\frac{\partial T}{\partial n} \, dS. \tag{3}$$

Since

$$\int_V \overline{\nabla}\cdot(N_i \; k\overline{\nabla}T) \, dV = \int_V (\overline{\nabla}N_i)\cdot(k\overline{\nabla}T)dv + \int_V N_i\overline{\nabla}\cdot(k\overline{\nabla}T)dV, \tag{4}$$

we can combine Eqs. 2, 3 and 4 to get

$$\int_V \rho c N_i \dot{T}\, dV + \int_V (\overline{\nabla} N_i)\cdot(k\overline{\nabla} T)\, dV = \int_S N_i k \frac{\partial T}{\partial n}\, dS. \qquad (5)$$

Each node of the solid region has a separate discretized linear equation calculated from Eq. 5 using the appropriate shape function $N_i$. Thus each of the volume integrals on the left hand side of Eq. 5 yields a square coefficient matrix in the assembled set of equations. Calculation of these matrices is a standard process. Details are given by Zienkiewicz [1].

The discretized set of equations takes the form

$$\underline{\underline{C}}\,\dot{\underline{T}} + \underline{\underline{Y}}\,\underline{T} = \underline{v}\ , \qquad (6)^{\checkmark}$$

where there is a term by term correspondence with Eq. 5. The real symmetric matrices $\underline{\underline{C}}$ and $\underline{\underline{Y}}$ represent, respectively, the thermal capacitance and thermal admittance. The elements of the vector $\underline{T}$ are nodal temperatures. The vector $\underline{v}$, discussed in the following section, depends upon the thermal boundary conditions.

In the present development piecewise constant material properties have been assumed, i.e., k, ρ and c are constant within each element, but may vary from element to element. Also two-dimensional isoparametric elements are used. Applicable equations are summarized in Appendix A.

---

$^{\checkmark}$In this text a double underline denotes a rectangular matrix and single underline denotes a column vector.

## B. THERMAL BOUNDARY CONDITIONS

Thermal boundary conditions affect only those scalar equations of Eq. 6 which correspond to boundary nodes. Accordingly, the vector $\underline{v}$ is sparse. Also, in a single problem it is common to have different thermal boundary conditions on individual portions of the boundary. In what follows the subvectors of $\underline{v}$ (distinguished by individual superscripts) which correspond to separate boundary conditions are treated individually.

### 1. Insulated

It is clear that for insulated boundary conditions the subvector $\underline{v}^{(1)}$ of the right hand side of Eq. 6 corresponding to this boundary condition is zero, since $\frac{\partial T}{\partial n} = 0$.

### 2. Convection

The heat transfer mechanism occurs in the interface of the solid and fluid. If the fluid temperature is $\theta$ and the heat transfer coefficient is h, then equating heat conducted away from the surface to the efflux from the solid [2] gives

$$-k \left(\frac{\partial T}{\partial n}\right)_S = h(T-\theta), \qquad (7)$$

where the subscript S means that the derivative is evaluated at the surface.

For constant h:

$$\int_S N_i \, k \frac{\partial T}{\partial n} \, dS = h \int_S N_i (\theta - T) \, dS. \qquad (8)$$

In what follows the fluid temperature $\theta$ is taken to be a specified function of position and time. For purposes of discretization, the fluid temperature is specified at a discrete number of fluid "nodes." If $\theta_j$ represents the fluid temperature at fluid node j, then the fluid temperature along the boundary may be represented by

$$\theta = \Sigma N_j \theta_j, \tag{9}$$

where the $N_j$ are one-dimensional forms of the shape functions used for the solid. Substitution in Eq. 8 gives

$$\int_S N_i \, k \, \frac{\partial T}{\partial n} \, dS = \sum_j y^*_{ij} (\theta_j - T_j), \tag{10}$$

where the summation extends over the surface nodes and the coefficients $y^*_{ij}$ are given by

$$y^*_{ij} = h \int_S N_i N_j \, dS. \tag{11}$$

Assembling the contributions from Eq. 10, the subvector $v^{(2)}$ for the convection boundary condition may be written

$$v^{(2)} = \underline{\underline{Y}}^* \underline{\theta}. \tag{12}$$

The contributions $-\Sigma\, y^*_{ij} T_j$ from Eq. 10 are included by augmenting the matrix $\underline{\underline{Y}}$ (see Eq. 13 below).

3. Constant Temperature

If $\theta$ represents the constant temperature desired at the wetted surface, then we can use the convection boundary condition and replace h by a big number (say $10^{20}$). Since h is very large, then for thermal equilibrium the temperature

T at the surface will be forced to equal θ. So the subvector $\underline{v}^{(3)}$ for the portion corresponding to the constant temperature boundary condition can be obtained from Eq. 12.

Upon the application of these boundary conditions in a single problem, the right-hand side vector $\underline{v}$ will be combined from the corresponding subvectors and the finite element discretized equation becomes

$$\underline{\underline{C}}\ \dot{\underline{T}} + \underline{\underline{Y}}^{+}\ \underline{T} = \underline{v}, \tag{13}$$

where $\underline{\underline{Y}}^{+} = \underline{\underline{Y}} + \underline{\underline{Y}}^{*}$.

## C. EXACT TIME SOLUTION WITH SPATIAL DISCRETIZATION

We consider only the solution of Eq. 13 for $\underline{v}$ = constant with $\underline{T} = \underline{a}$ at time $\tau = 0$. Let $\underline{T}_s$ be a particular solution (steady state) with $\dot{\underline{T}}_s = 0$ so that

$$\underline{T}_s = (\underline{\underline{Y}}^{+})^{-1}\ \underline{v}. \tag{14}$$

For the homogeneous equation

$$\underline{\underline{C}}\ \dot{\underline{T}} + \underline{\underline{Y}}^{+}\ \underline{T} = 0 \tag{15}$$

the assumption $\underline{T} = \underline{w}\exp(-\lambda\tau)$, where $\underline{w}$ is a vector and $\lambda$ is a scalar, yields the form

$$\underline{\underline{Y}}^{+}\ \underline{w} = \lambda\ \underline{\underline{C}}\ \underline{w}\ . \tag{16}$$

It is apparent that Eq. 16 defines an eigenvalue problem. Let $\underline{\underline{\Lambda}}$ be the spectral matrix and $\underline{\underline{W}}$ the modal matrix with normalization according to

$$\underline{\underline{W}}^{T}\ \underline{\underline{C}}\ \underline{\underline{W}} = \underline{\underline{I}}\ , \tag{17}$$

where $\underline{\underline{I}}$ is the identity matrix of the same order as $\underline{\underline{C}}$. Now let

$$\underline{T} = \underline{\underline{W}} \exp(-\underline{\underline{\Lambda}}\tau)\ \underline{b} \tag{18}$$

where $\underline{b}$ is a constant vector. Substituting this in Eq. 15 gives

$$\underline{\underline{Y}}^{+}\ \underline{\underline{W}} \exp(-\underline{\underline{\Lambda}}\tau)\ \underline{b} = \underline{\underline{C}}\ \underline{\underline{W}}\ \underline{\underline{\Lambda}} \exp(-\underline{\underline{\Lambda}}\tau)\ \underline{b}. \tag{19}$$

Now Eq. 19 is satisfied for all $\underline{b}$ if

$$\underline{\underline{W}}^{T}\ \underline{\underline{Y}}^{+}\ \underline{\underline{W}} = \underline{\underline{\Lambda}}\ , \tag{19'}$$

and this is guaranteed to be satisfied since $\underline{\underline{\Lambda}}$ and $\underline{\underline{W}}$ are spectral and modal matrices for the eigenvalue problem of Eq. 16 with $\underline{\underline{W}}$ normalized according to Eq. 17.

Returning to the original problem (Eq. 13), the complete solution may be written as

$$\underline{T} = \underline{\underline{W}}\ (\underline{\beta} + \exp(-\underline{\underline{\Lambda}}\tau)\ \underline{b}) \tag{20}$$

where

$$\underline{T}_s = \underline{\underline{W}}\ \underline{\beta}\ , \tag{21}$$

and $\underline{\beta}$ is a constant vector. Now $\underline{\beta}$ may be found (using Eq. 14) to be

$$\underline{\beta} = \underline{\underline{\Lambda}}^{-1}\ \underline{\underline{W}}^{T}\underline{v}\ . \tag{22}$$

Substituting this result into Eq. 20 and using the initial condition gives the result

$$\underline{b} = \underline{\underline{W}}^{-1}\ \underline{a} - \underline{\underline{\Lambda}}^{-1}\ \underline{\underline{W}}^{T}\ \underline{v}\ . \tag{23}$$

The general solution of Eq. 13 may thus be written as

$$\underline{T} = \underline{\underline{W}}(\underline{\underline{I}} - \exp(-\underline{\underline{\Lambda}}\tau))\, \underline{\underline{\Lambda}}^{-1}\, \underline{\underline{W}}^{T}\underline{v} + \underline{\underline{W}}\, \exp(-\underline{\underline{\Lambda}}\tau)\underline{\underline{W}}^{-1}\underline{a}. \quad (24)$$

For purposes of the present program non-zero components of vector $\underline{v}$ are to be specified as piecewise linear functions of time. During each segment of time history of $\underline{v}$ an analytical solution of Eq. 13 is possible in the form of a particular solution plus a complementary solution such as Eq. 20. At each node the corresponding time variation of temperature will consist of a linear part contributed by the particular solution and a sum of n terms representing the complementary part. Each of these n terms decays exponentially with a separate time constant. In principle it is a straightforward process to find each particular solution and accompanying complementary solution.

Contemplated problems may typically have from 10 to 40 segments required to represent the piecewise linear variation of $\underline{v}$. The number of body nodes n will be of the order of 100 or more. In view of the number of particular solutions required, each accompanied by an individual complementary solution of the form given by Eq. 20, it was concluded that a numerical solution of Eq. 13 would be considerably more economical than an analytic one such as that given by Eq. 24.

## D. TIME INTEGRATION

In this section the relative merits of the Runge-Kutta and trapezoidal methods of time integration are discussed. Since either of these methods will give an exact result if the solution is a linear function of time, investigation

is confined to performance on a single scalar equation

$$\dot{y} + \lambda y = 0 \tag{25}$$

whose solution $y = y_0 \exp(-\lambda\tau)$ is of the same form as the components of the complementary solution (Eq. 20 ).

1. Runge-Kutta Method

A method introduced by Runge and subsequently elaborated by Heun and Kutta [3] is widely used for the numerical solution of first order ordinary differential equations. This algorithm prescribes a sequence of calculations for determining the ordinate $y_{i+1}$ at time $\tau_{i+1} = \tau_i + \Delta\tau$ in terms of $y_i$ and values of $\dot{y}$ at intermediate and end points of the interval $\Delta\tau$. The fourth-order form, which requires four evaluations of $\dot{y}$, gives for Eq. 25 the result

$$\frac{y_{i+1}}{y_i} = 1 - \lambda\Delta\tau + \frac{(\lambda\Delta\tau)^2}{2!} - \frac{(\lambda\Delta\tau)^3}{3!} + \frac{(\lambda\Delta\tau)^4}{4!} . \tag{26}$$

The right-hand side of Eq. 26 represents the first five terms of the Taylor expansion of the exact solution $(y_{i+1}/y_i = \exp(-\lambda\Delta\tau))$ so we may conclude that the relative error in each time step is less than modulus of the next term: $(\lambda\Delta\tau)^5/5!$.

In addition to providing the apparent prospect for high precision indicated by this error bound, the Runge-Kutta method also permits changes of time increment during the integration process without requiring additional computationally expensive matrix decompositions. The attractiveness of these two features dictated a thorough exploration of the potential of this method for the present application. The

is confined to performance on a single scalar equation

$$\dot{y} + \lambda y = 0 \tag{25}$$

whose solution $y = y_o \exp(-\lambda\tau)$ is of the same form as the components of the complementary solution (Eq. 20 ).

1. Runge-Kutta Method

A method introduced by Runge and subsequently elaborated by Heun and Kutta [3] is widely used for the numerical solution of first order ordinary differential equations. This algorithm prescribes a sequence of calculations for determining the ordinate $y_{i+1}$ at time $\tau_{i+1} = \tau_i + \Delta\tau$ in terms of $y_i$ and values of $\dot{y}$ at intermediate and end points of the interval $\Delta\tau$. The fourth-order form, which requires four evaluations of $\dot{y}$, gives for Eq. 25 the result

$$\frac{y_{i+1}}{y_i} = 1 - \lambda\Delta\tau + \frac{(\lambda\Delta\tau)^2}{2!} - \frac{(\lambda\Delta\tau)^3}{3!} + \frac{(\lambda\Delta\tau)^4}{4!} . \tag{26}$$

The right-hand side of Eq. 26 represents the first five terms of the Taylor expansion of the exact solution $(y_{i+1}/y_i = \exp(-\lambda\Delta\tau))$ so we may conclude that the relative error in each time step is less than modulus of the next term: $(\lambda\Delta\tau)^5/5!$.

In addition to providing the apparent prospect for high precision indicated by this error bound, the Runge-Kutta method also permits changes of time increment during the integration process without requiring additional computationally expensive matrix decompositions. The attractiveness of these two features dictated a thorough exploration of the potential of this method for the present application. The

disqualifying defect which emerged after studying a number of examples is readily appreciated from examination of Table I. For values of $\lambda\Delta\tau$ less than 0.5 it is apparent that Runge-Kutta scheme affords acceptable engineering accuracy. However, when the method is applied to solution of Eq. 13 we must deal with a number of $\lambda$'s equal to the number of nodes (see Eq. 20 ). This number may be greater than 100 and the ratio of the largest $\lambda$ to the smallest may easily exceed 1000. Although the solution is dominated by the contributions of the eigenvectors corresponding to the smaller $\lambda$s, it is clear that the solution will be unstable if the largest $\lambda\Delta\tau$ exceeds about 2.7. Because an unacceptably small $\Delta\tau$ is required in typical problems, the Runge-Kutta method was rejected.

2. Trapezoidal Method

The trapezoidal method estimates $y_{i+1}$ from the formula

$$y_{i+1} = y_i + \frac{\Delta\tau}{2} (\dot{y}_i + \dot{y}_{i+1}). \tag{27}$$

Substituting for $\dot{y}_i$ and $\dot{y}_{i+1}$ from Eq. 25 and rearranging gives

$$\frac{y_{i+1}}{y_i} = \frac{2 - \lambda\Delta\tau}{2 + \lambda\Delta\tau} . \tag{28}$$

Series expansion of the right-hand side provides an error bound (per step):

$$(\lambda\Delta\tau)^3/12.$$

From the point of view of the size of $\lambda\Delta\tau$ this method has no stability limit, but has slow attenuation with alteration

TABLE I

ATTENUATION FACTOR COMPARISON

Fourth Order Runge-Kutta Algorithm

| $\lambda\Delta\tau$ | $y_{i+1}/y_i$ (Runge-Kutta) | $\exp(-\lambda\Delta\tau)$ (Exact) |
|---|---|---|
| .0001 | .9999 | .9999 |
| .001 | .9990 | .9990 |
| .01 | .9901 | .9901 |
| .1 | .9048 | .9048 |
| .2 | .8187 | .8187 |
| .5 | .6068 | .6065 |
| 1.0 | .3750 | .3679 |
| 2.0 | .3333 | .1353 |
| 2.5 | .6484 | .0821 |
| 3.0 | 1.3750 | .0498 |
| 4.0 | 5.0000 | .0183 |
| 8.0 | 110.3333 | .0003 |
| 10.0 | 291.0000 | .0000 |
| 20.0 | 5514.3333 | .0000 |
| 50.0 | 240784.3333 | .0000 |
| 100.0 | 4004901.0000 | .0000 |

in sign for large $\lambda\Delta\tau$. Table II shows this behavior.

Since a wide usable range of $\lambda\Delta\tau$ is essential and the stability of trapezoidal integration is guaranteed, this method is chosen for the present program.

Applying the trapezoidal algorithm to Eq. 13 yields

$$\underline{\underline{A}}\ \underline{T}^{i+1} = \underline{\underline{G}}\ \underline{T}^{i} + \frac{\Delta\tau}{2}\left(\underline{v}^{i+1} + \underline{v}^{i}\right), \qquad (29)$$

where

$$\underline{\underline{A}} = \underline{\underline{C}} + \frac{\Delta\tau}{2}\underline{\underline{Y}}^{+},$$

$$\underline{\underline{G}} = \underline{\underline{C}} - \frac{\Delta\tau}{2}\underline{\underline{Y}}^{+}.$$

and the superscripts denote evaluation at discrete time intervals $\Delta\tau$. If m is the order of the capacitance and admittance matrices, $\underline{\underline{C}}$ and $\underline{\underline{Y}}$, then once a certain step size $\Delta\tau$ is chosen, it requires $m^3/3$ operations to perform the needed triangular decomposition of $\underline{\underline{A}}$. Thus, for large m, a change of step size $\Delta\tau$ becomes costly from the point of view of computer time. Accordingly, in the present program only one time step size is used throughout each problem.

Also, for assuring sufficiently rapid attenuation of the components corresponding to the large $\lambda\Delta\tau$, the following correction is utilized.

3. Irons' Correction

Irons proposed a scheme [4] for augmenting the attenuation of the contributions of those eigenvectors for which $\lambda\Delta\tau$ is large. Define

TABLE II

ATTENUATION FACTOR COMPARISON

Trapezoidal Integration

| $\lambda\Delta\tau$ | $y_{i+1}/y_i$ (Trapezoidal) | $e^{-\lambda\Delta\tau}$ (Exact) |
|---|---|---|
| .0001 | .9999 | .9999 |
| .001 | .9990 | .9990 |
| .01 | .9901 | .9901 |
| .1 | .9048 | .9048 |
| .2 | .8182 | .8187 |
| .3 | .7391 | .7408 |
| .5 | .6000 | .6065 |
| 1.0 | .3333 | .3679 |
| 2.0 | .0000 | .1353 |
| 2.5 | -.1111 | .0821 |
| 3.0 | -.2000 | .0498 |
| 4.0 | -.3333 | .0183 |
| 8.0 | -.6000 | .0003 |
| 10.0 | -.6667 | .0000 |
| 20.0 | -.8182 | .0000 |
| 50.0 | -.9231 | .0000 |
| 100.0 | -.9608 | .0000 |

$$y_i^* = .25\, y_{i-1} + .5\, y_i + .25\, y_{i+1}, \tag{30}$$

where $y_i$ and $y_{i+1}$ are obtained from $y_{i-1}$ by trapezoidal integration.

In the program presented in Appendix B, Eq. 30 is used after every 10 steps of time integration. Table III shows the resulting modifications.

## E. ESTIMATION OF EXTREME EIGENVALUES

Analytic results for one-dimensional heat conduction give, for an eigenvalue,

$$\lambda = \frac{\pi^2 k}{4\rho c} \frac{1}{d^2}, \tag{31}$$

where d is the distance between points of extreme temperature and zero temperature.

If we use this to estimate the smallest $\lambda$ in cylindrical coordinates, two modifications are recommended.

1. Assume that the point of zero temperature is in the fluid at a distance from the wall equal to $k/h$, where h is the surface heat transfer coefficient.
2. If there are two approximately orthogonal paths for heat flow from the (single) maximum temperature point, then replace $1/d^2$ in the above formula by

$$\frac{1}{d^2} = \frac{1}{d^2_{\text{min.}}} + \frac{1}{d^2_{\text{max.}}}. \tag{32}$$

For estimating the largest $\lambda$, the surface heat transfer coefficient has no significant effect. We may continue to

TABLE III

EFFECTS OF USING IRONS' CORRECTION
AFTER 10 STEPS OF INTEGRATION

| $\lambda\Delta\tau$ | $\exp(-10\lambda\Delta\tau)$ Exact | $y_{10}/y_0$ Trapezoidal | $y^*_{10}/y_0$ Corrected |
|---|---|---|---|
| 0.00010 | 0.99900 | 0.99900 | 0.99900 |
| 0.00020 | 0.99800 | 0.99800 | 0.99800 |
| 0.00100 | 0.99005 | 0.99005 | 0.99005 |
| 0.01000 | 0.90484 | 0.90484 | 0.90486 |
| 0.10000 | 0.36788 | 0.36757 | 0.36849 |
| 0.20000 | 0.13534 | 0.13443 | 0.13579 |
| 0.30000 | 0.04979 | 0.04866 | 0.04978 |
| 0.50000 | 0.00674 | 0.00605 | 0.00645 |
| 1.00000 | 0.00005 | 0.00002 | 0.00002 |
| 2.00000 | 0.00000 | 0.00000 | 0.00000 |
| 4.00000 | 0.00000 | 0.00002 | -0.00001 |
| 8.00000 | 0.00000 | 0.00605 | -0.00040 |
| 10.00000 | 0.00000 | 0.01734 | -0.00072 |
| 20.00000 | 0.00000 | 0.13443 | -0.00136 |
| 50.00000 | 0.00000 | 0.44914 | -0.00072 |
| 100.00000 | 0.00000 | 0.67028 | -0.00027 |

use the same formula for $\frac{1}{d^2}$ , but now consider only the smallest element and take

$$d_{min} = \frac{\text{length of smallest side}}{4},$$
$$d_{max} = \frac{\text{length of largest side}}{4}. \qquad (32')$$

A comparison of estimates based on Eq. 31, 32, 32' with the exact solution of the eigenvalue problem has been carried out for several examples. Based on these comparisons, it is believed that these estimates are sufficiently accurate for choosing a time step and estimating the time of occurrence of the extreme stresses.

## III. ONE-DIMENSIONAL HEAT CONDUCTION

For comparison of numerical (time) integration methods, studies of one-dimensional heat conduction were made. In this section numerical results for trapezoidal time integration using Irons' correction are compared with the exact transient temperature solution.

Consider a flat slab of thickness L with conductivity k, density $\rho$, specific heat c, zero initial temperature, one face insulated and the other in contact with fluid at temperature $T_f$ (Fig. 1). The surface heat transfer is h. The exact transient heat conduction solution is available [5] as

$$T = T_f \left\{ 1 - 2 \sum_{i=1}^{\infty} \frac{\text{Sin}\mu_i L \ \text{Cos}\mu_i x}{\mu_i L + \text{Sin}\mu_i L \ \text{Cos}\mu_i L} \cdot e^{-\mu_i^2 \frac{k}{\rho c} \tau} \right\} \quad (33)$$

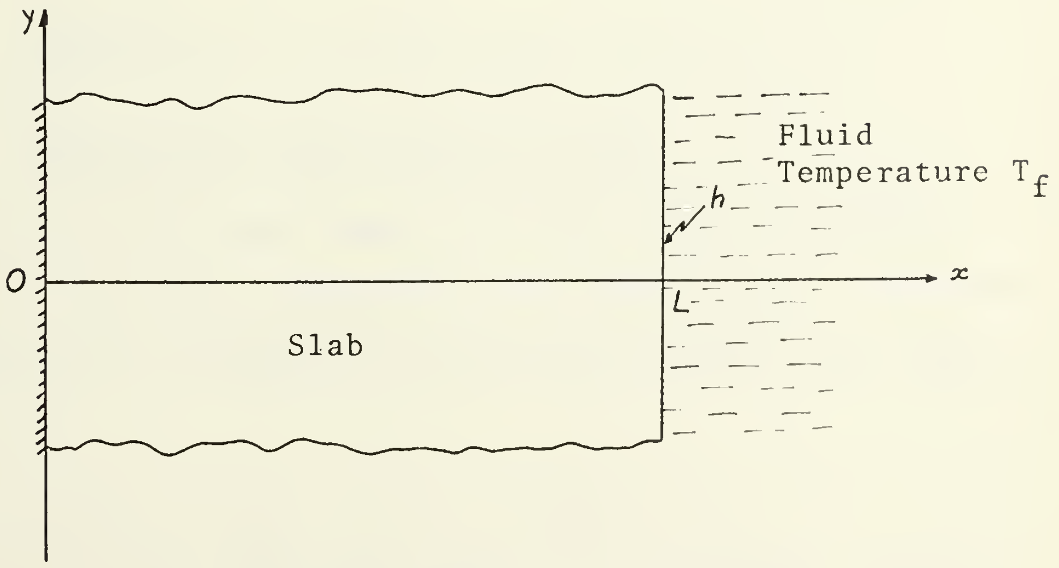


Figure 1. Slab with One Face Insulated and Another in Contact with Fluid.

where $k/\rho c$ is the diffusivity and the $\mu_i$ are the solutions of the transcendental equation

$$\text{Tan } \mu L = \frac{hL}{k} \; \frac{1}{\mu L} \; . \tag{34}$$

For the finite element comparisons we subdivide the distance L into m one-dimensional 3-noded elements and use the corresponding isoparametric shape functions. (The working equations, shape functions and the element capacitance and admittance matrices are given in Appendix A.)

In the course of this investigation separate computer programs were written to evaluate nodal transient temperatures using the following methods:

(a) exact transient temperature solution, Eq. 33;

(b) exact time solution with spatial discretization (section II.C, Eq. 24);

(c) Runge-Kutta time integration (section II.D, part a);

(d) trapezoidal time integration (section II.D, part b);

(e) trapezoidal time integration with Irons' correction (section II.D, part c).

For an initial step change of fluid temperature from zero to 1, transient temperatures have been found. For these comparisons the parameters (in consistent units) were taken to be:

$$L = 8, \; \rho = 25, \; k = 8, \; c = 5 \text{ and } h = 5$$

The distance L was subdivided into m = 3 elements. For the present purpose, comparison is confined to the exact solution (item (a)), and the finally adopted system (item (e)).

In Table IV, temperatures at the two faces (x = 0, x = 8) are compared for various times. The trapezoidal integration has been performed using the constant time increment unity and the Irons' correction is applied after every 10 increments.

It is believed that Table IV demonstrates that the numerical integration method gives adequate accuracy for engineering applications.

TABLE IV

COMPARISON OF ONE-DIMENSIONAL TRANSIENT TEMPERATURES

| Method | x | Time=0 | Time=10 | Time=20 | Time=30 | Time=50 | Time=70 |
|---|---|---|---|---|---|---|---|
| Exact | 0. | .00000 | .00000 | .00000 | .00002 | .00095 | .00560 |
| Trapezoidal with Irons'✓ | 0. | .00000 | .00028 | .00007 | .00066 | .00207 | .00619 |
| Exact | 8. | .00000 | .38431 | .47684 | .53284 | .60264 | .64685 |
| Trapezoidal with Irons'✓ | 8. | .00000 | .38720 | .47716 | .53262 | .60256 | .64686 |

✓Irons' correction is used after every 10 steps of trapezoidal integration.

# IV. STRESS PROBLEM

For bodies of revolution deformed symmetrically under axisymmetric loading, the stress distribution is two-dimensional. Since deformation is symmetric about the axis of revolution, cylindrical coordinates $(R, Z, \theta)$ are used. It follows that the stress components are independent of the angle $\theta$ and all derivatives with respect to $\theta$ are zero. Also the components of shearing stress $\tau_{R\theta}$ and $\tau_{Z\theta}$ vanish on account of the symmetry. But since any radial displacement induces a strain $\varepsilon_\theta$ in the circumferential direction, this non-zero component of strain and the three in-plane components $(\varepsilon_Z, \varepsilon_R, \gamma_{RZ})$, complete the state of strain at a point in any axisymmetric situation. Hence the state of stress for an axisymmetric body under axisymmetric loading is given by

$$\underline{\sigma} = \langle \sigma_Z \ \sigma_R \ \sigma_\theta \ \tau_{RZ} \rangle^T \ . \qquad (35)$$

In this chapter the stiffness matrix of an axisymmetric body and the thermal, pressure, and centrifugal load vectors are formulated and, finally, evaluation of stresses at a point is discussed. The treatment closely follows that of Zienkiewicz [1] and this reference should be consulted for further details.

## A. STIFFNESS MATRIX

The elements used are bodies of revolution (about the Z axis). For analysis it is sufficient to describe the cross-section in the R,Z plane. In Fig. 2 such an element and the local $\xi,\eta$ coordinates are shown.

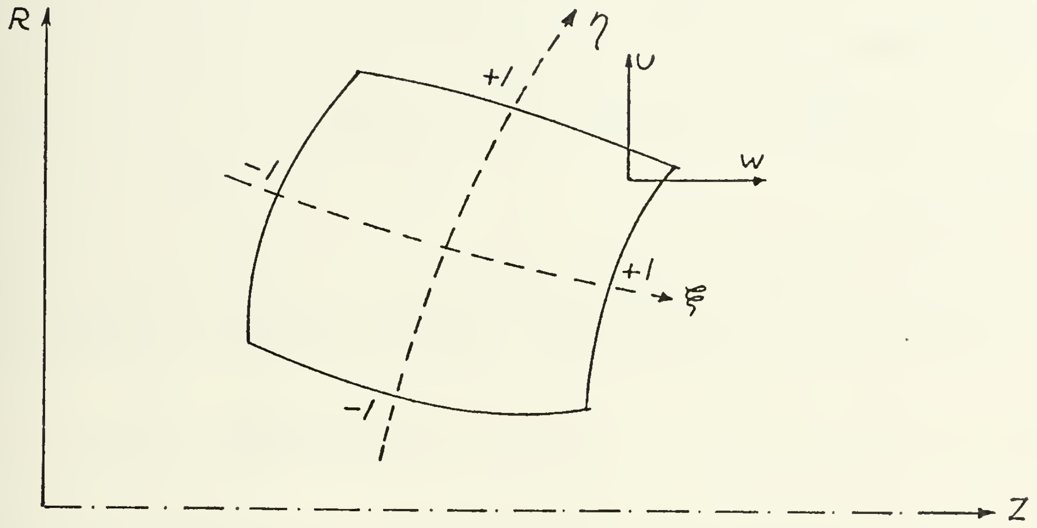


Figure 2. Quadrilateral Element Representation.

If u and w are the displacement components at a point in the directions of R and Z respectively, then these displacement components may be defined in terms of the nodal displacements by the appropriate isoparametric shape functions as

$$u = \sum_{i=1}^{8} N_i u_i \ , \qquad w = \sum_{i=1}^{8} N_i w_i \ , \tag{36}$$

where $N_i$, a function of $\xi$ and $\eta$ , is the shape function for element node i and $u_i, w_i$, are the nodal displacement components. The strain-displacement relations [6] can now be used to obtain the components of the strain vector. Thus

$$\underline{\varepsilon} = \underline{\underline{B}}\ \underline{\delta}\ , \tag{37}$$

where $\underline{\underline{B}}$ is the standard rectangular strain-displacement matrix of any finite element formulation, a function of the local coordinates $\xi$ and $\eta$, and $\underline{\delta}$ is the vector of nodal displacement. (See Appendix A, part 3, where the applicable formulas and useful equations are summarized).

If the elasticity matrix for an isotropic material is $\underline{\underline{D}}$, then the stress vector $\underline{\sigma}$ at a point is given by

$$\underline{\sigma} = \underline{\underline{D}}\ \underline{\varepsilon}\ . \tag{38}$$

Now, by evaluation of the total strain energy in the element, the element stiffness matrix can readily be obtained as

$$\underline{\underline{K}}^e = \int \underline{\underline{B}}^T\ \underline{\underline{D}}\ \underline{\underline{B}}\ dV\ , \tag{39}$$

where the integration extends over the volume of the element.

In the present program the upper triangle of each element stiffness matrix is evaluated by numerical integration using four Gauss points within the range of $\xi$ and of $\eta$ [1]. The element contribution is immediately placed in the system stiffness matrix, which is stored in banded form.

## B. THERMAL LOAD VECTOR

If we denote T as the difference between local temperature and reference temperature, then the thermal strain $\underline{\varepsilon}_o$ is given as

$$\underline{\varepsilon}_o = \underline{U}\alpha T, \tag{40}$$

where

$$\underline{U} = \langle 1\ 1\ 1\ 0\rangle^T$$

and $\alpha$ is the coefficient of thermal expansion.

The thermal load vector $\underline{F}_T^e$ is given by Zienkiewicz [1] as

$$\underline{F}_T^e = \int \underline{\underline{B}}^T \underline{\underline{D}}\ \underline{\varepsilon}_o^e\ dV\ . \tag{41}$$

From the point of view of the numerical evaluation it is interesting to note, however, that the product $\underline{\underline{D}}\ \underline{\varepsilon}_o^e$ in Eq. 41 will reduce to

$$\underline{\underline{D}}\ \underline{\varepsilon}_o^e = \frac{E\,\alpha\,T}{1-2\nu}\ \underline{U}\ , \tag{42}$$

where E is the modulus of elasticity and $\nu$ is Poisson's ratio. Thus

$$\underline{F}_T^e = \frac{E\alpha}{1-2\nu} \int T\ \underline{\underline{B}}^T\ \underline{U}\ dV\ , \tag{43}$$

where $T = \sum_{i=1}^{n} N_i T_i$ and n is the number of nodes in each element.

In the attached computer program in Appendix B the advantage of the simplicity of Eq. 42 has been utilized. Also, since $\underline{\underline{B}}$ has some zero components, in the process of multiplication of $\underline{\underline{B}}^T\ \underline{U}$, simply the addition of the appropriate non-zero components of each column of the $\underline{\underline{B}}^T$ has been performed. Finally, Eq. 43 has been integrated with four Gauss points.

## C. PRESSURE LOAD VECTOR

Consider a quadrilateral element as in Fig. 3 on the boundary of the axisymmetric cross-section where constant pressure P is applied. The infinitesimal force dF due to the normal pressure acting on the inner infinitesimal circumferential surface dS is

$$dF = P\,dS = 2\pi RP\,d\ell \,, \tag{44}$$

where $d\ell$ is the infinitesimal length along the side of quadrilateral.

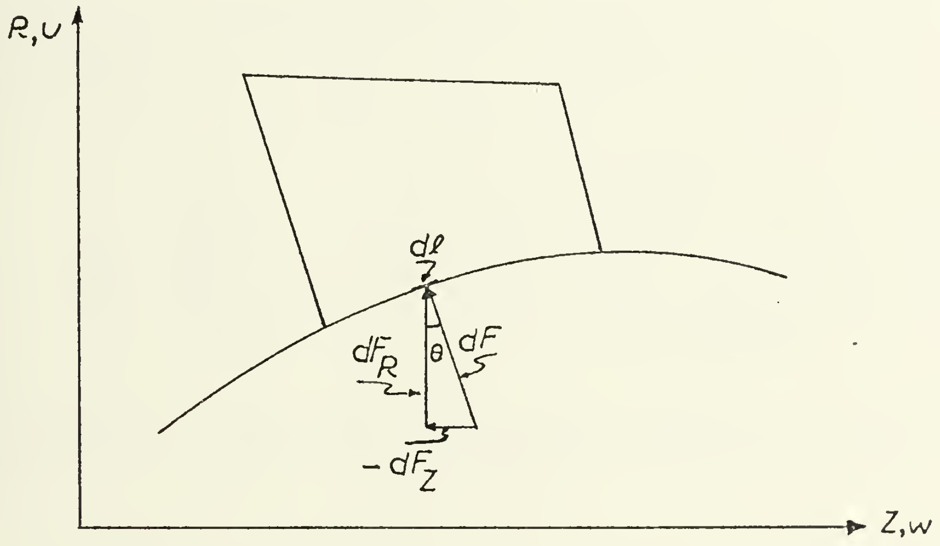


Figure 3. Boundary Element Under Pressure.

Let $F_R$ and $F_Z$ be the components of the pressure force in the R and Z directions respectively, then

$$dF_R = dF\ \mathrm{Cos}\ \theta, \quad dF_Z = -dF\ \mathrm{Sin}\ \theta \,, \tag{45}$$

where $\mathrm{Sin}\ \theta = \frac{dR}{d\ell}$ and $\mathrm{Cos}\ \theta = \frac{dZ}{d\ell}$. Therefore

$$dF_R = (2\pi RP)dZ \,, \quad dF_Z = -(2\pi RP)dR. \tag{46}$$

Since the total work W done by the normal force F is equal to the sum of the work done in R and Z directions,

$$W = \int u \, dF_R + \int w \, dF_Z$$

$$= 2\pi P \, (\int R \, u \, dZ - \int R \, w \, dR). \qquad (47)$$

Now, by using the appropriate shape functions, each component of Eq. 47 may be defined in terms of the nodal values. Since

$$W = (\underline{\delta}^e)^T \, \underline{F}_p^e, \qquad (48)$$

where $\underline{F}_p^e$ is the element pressure load vector vector contributed by node j is explicitly given as

$$F_{j_P}^e = \begin{bmatrix} F_R \\ F_Z \end{bmatrix}_{j_p}^e = 2\pi P \int_{-1}^{1} (\Sigma N_i R_i) \begin{bmatrix} \Sigma \dfrac{\partial N_i}{\partial \xi} Z_i \\ -\Sigma \dfrac{\partial N_i}{\partial \xi} R_i \end{bmatrix} N_j \, d\xi \ , \qquad (49)$$

where $N_j$ in Eq. 49 is the appropriate shape function for node j.

## D. CENTRIFUGAL LOAD VECTOR

Refer again to Fig. 2 and assume that the body is rotating about the Z axis with angular speed $\Omega$. Then the centrifugal force per unit volume at a point distant R from the Z axis will be $\rho\Omega^2 R$, where $\rho$ is the density of the material. The work done in this case is

$$W = \int_V \rho\Omega^2 \, R \, u \, dV \ . \qquad (50)$$

For constant $\rho\Omega^2$ the corresponding element centrifugal load vector is readily obtained by evaluation of the components of the integral in Eq. 50 in terms of the nodal variables, i.e.,

$$F^e_{j_c} = 2\pi\rho\Omega^2 \int_{-1}^{1} \int_{-1}^{1} (\Sigma N_i R_i)^2 \det \underline{\underline{J}}\, N_j \, d\xi d\eta, \qquad (51)$$

where det $\underline{\underline{J}}$ is the determinant of the Jacobian coordinate transformation matrix (see Appendix A).

## E. STRUCTURAL BOUNDARY CONDITIONS

The structural boundary conditions implemented in the program are:

(a) one or more nodes prevented from moving axially;

(b) one or more nodes prevented from moving radially;

(c) the right-hand end cross-sectional plane remains plane and the transmitted axial force is specified. Hereinafter this will be referred to as the plane-end boundary condition.

The computer program presented in Appendix B has the capability of handling any combination of the above mentioned structural boundary conditions. For boundary conditions of the types (a) and (b), simply multiplying the corresponding diagonal component of the stiffness matrix by $10^{20}$ gives zero displacement [8] (for practical purposes). For the boundary condition of type (c) both ends are initially fixed axially for all solutions. An additional solution is obtained for unit axial displacement of one end. The axial force is evaluated for each solution. Superposition is performed by adding the displacement vectors for the given

loadings (thermal plus mechanical), plus an appropriate fraction of the vector found for unit axial displacement. This fraction is chosen so that the resultant axial load has the specified value.

## F. SYSTEM EQUATION SOLVER

Once the desired structural boundary conditions are applied, then the problem is to find the nodal displacement vector $\underline{\delta}$, corresponding to a given number of load vectors. We have

$$\underline{\underline{K}}\ \underline{\delta} = \underline{F}\ , \tag{52}$$

where $\underline{\underline{K}}$ is the system stiffness matrix in banded form and $\underline{F}$ is a load vector. In the present computer program a single Cholesky decomposition is performed on $\underline{\underline{K}}$. Then, by a process of forward and back substitution, each load vector is replaced by the corresponding displacement vector.

## G. PRINCIPLE OF SUPERPOSITION

Upon the evaluation of the displacement vectors due to the various types of loading, the principle of superposition can be applied on the displacement vectors. On each thermal displacement vector the displacement due to any other type of loading is superimposed and, as the result, the number of displacement vectors is reduced to the number of thermal load vectors.

## H. STRESS EVALUATION

From the system displacement vector $\underline{\delta}$, the displacement vector of each element $\underline{\delta}^e$ may be obtained easily. Then the

corresponding element strain vector $\underline{\varepsilon}^e$ at any point can be found from

$$\underline{\varepsilon}^e = \underline{\underline{B}}\, \underline{\delta}^e \, . \tag{53}$$

Finally, the corresponding element stress vector $\underline{\sigma}^e$ is obtained by

$$\underline{\sigma}^e = \underline{\underline{D}}\, (\underline{\varepsilon}^e - \underline{\varepsilon}_o^e) \, . \tag{54}$$

Since normally the stresses on the inner and outer surfaces of axisymmetric bodies are desired, in the computer program presented in the Appendix B provision has been made to calculate the stresses at the two Gauss points corresponding to $\xi = \pm \frac{1}{\sqrt{3}}$ on the inner and outer boundaries of each element (where $\eta = \pm 1$).

Upon the evaluation of the stresses at each point the mean stress and the octahedral shearing stress [7] are calculated. The program gives as output the extreme values of these stresses, the R and Z coordinates of the corresponding points, and the times of occurrence.

## V. ONE-DIMENSIONAL TRANSIENT STRESSES

In this section a one-dimensional comparison of stresses is made between exact and finite element results. The transient temperature problem is the one previously described in Section III.

If the slab edges are free to translate in the plane of the slab, but are prevented from rotating, the exact solution for thermal stress [9] is

$$\sigma_y = \sigma_z = \frac{E\alpha}{1-\nu}(T_{avg} - T) , \qquad (55)$$

where $T$ is the local temperature (Eq. 33), and

$T_{avg}$ is the average temperature in the slab

If we choose $E = 2$, $\alpha = .50$, and $\nu = 0$ (all in consistent units), then the maximum stress obtained from the exact solution is

$$\sigma_{max} = -.477786$$

and it occurs at $x = 8$, $\tau = 73$.

Using the finite element technique with trapezoidal time integration and Irons' correction every 10 steps, the maximum stress is found to be

$$\sigma_{max} = -.477778$$

and it also occurs at $x = 8$, $\tau = 73$, as before.

It is observed that the method chosen gives excellent results.

# VI. TEST PROBLEMS

Program integrity and accuracy have been verified by solving a number of test problems. Since stresses, whose evaluation depends upon derivatives of displacements, are known to be less accurate than temperatures, comparisons with exact results are confined to stresses. Individual problems are described below.

## A. THICK CYLINDER

Consider a thick cylinder with inside radius 30 inches and outside radius 50 inches and the following material properties.

| | | |
|---|---|---|
| Modulus of elasticity | $E = 28.9 \times 10^6$ | Psi |
| Poisson's ratio | $\nu = .28$ | - |
| Coefficient of thermal expansion | $\alpha = 7.22 \times 10^{-6}$ | 1/F° |
| Thermal conductivity | $k = 28.$ | $\frac{\text{Btu}}{\text{hr.ft°F}}$ |
| Density | $\rho = 489.$ | $\text{Lbm/ft}^3$ |
| Specific heat | $c = .111$ | $\frac{\text{Btu}}{\text{Lbm.°F}}$ |

An arbitrary length of 25 inches has been selected for the cylinder and it has been subdivided into two different element representations as in Figs. 4 and 5. The plane-end boundary condition with zero axial force is used. The stresses for various types of loading are compared with the corresponding exact analytic solutions as described below.

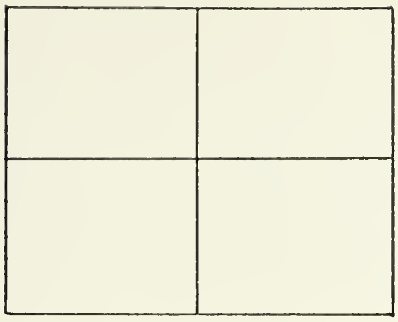


Figure 4. Two Radial Elements Representation of Thick Cylinder.

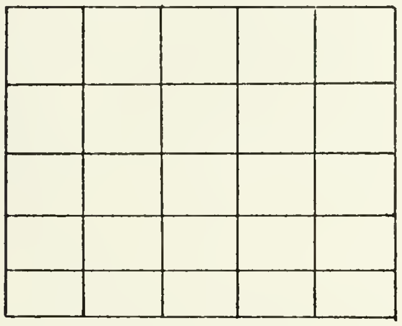


Figure 5. Five Radial Element Representation of Thick Cylinder.

1. Thermal Loading

For a linear variation of the temperature T = 20R the stresses obtained by the finite element method for the above representations are compared with the exact analytic solution in Fig. 6. The $\tau_{RZ}$ for this problem clearly is zero and the one obtained by the program was $10^{-9}$. The accuracy of the other results is clearly satisfactory.

2. Pressure Loading

A uniform pressure of 1000 psi. acts on the inner surface of the cylinder. Again, $\tau_{RZ}$ is zero and the program gives $10^{-10}$. In Fig. 7 the other stresses induced by this uniform pressure are compared with the exact solutions. Here also the accuracy of the results, even with two radial elements, is adequate.

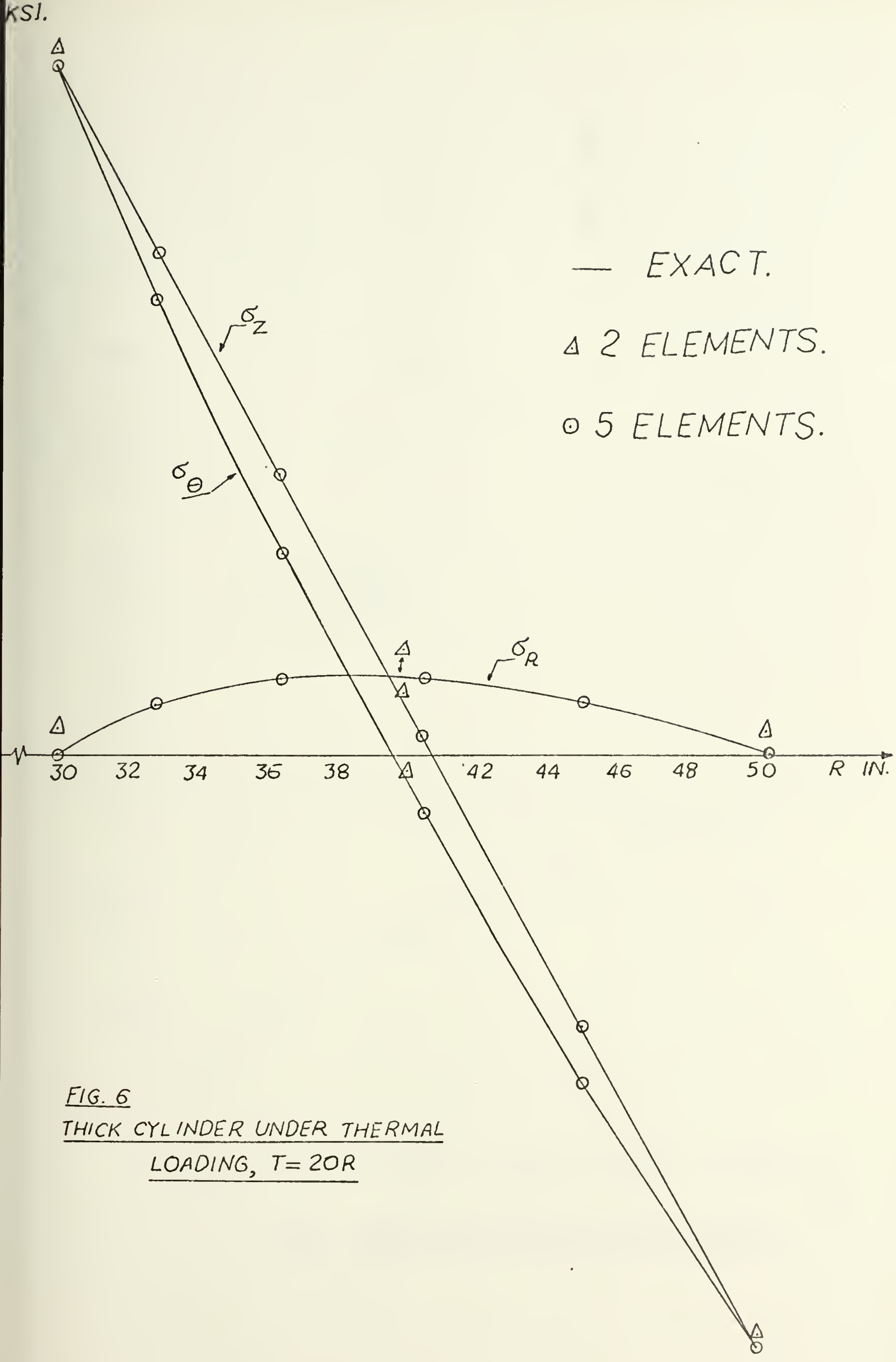


FIG. 6

THICK CYLINDER UNDER THERMAL LOADING, T= 20R

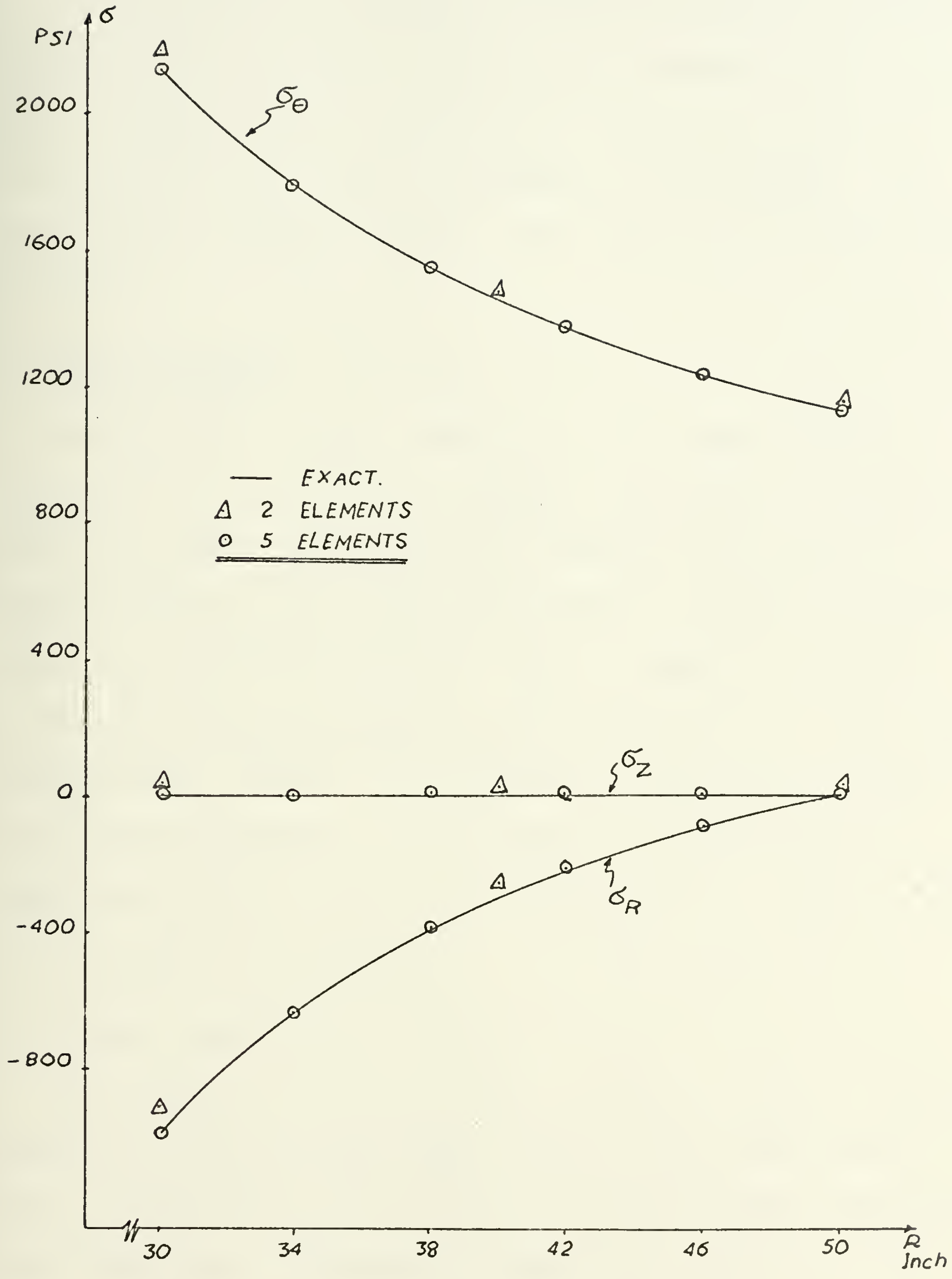


FIG. 7 THICK CYLINDER UNDER INTERNAL PRESSURE.

### 3. Centrifugal Loading

The speed of rotation has been assumed to be 500 revolutions per minute. The stresses obtained by the program are compared with the analytic solution in Fig. 8. In this case also the results obtained by the program, even with only two radial elements, are very close to the exact solution.

## B. HOLLOW SPHERE

We consider a hollow sphere with the same material properties as in the thick cylinder with inside spherical radius 30 inches and outside 50 inches. The loading is thermal with T = 20 ·(spherical radius). Symmetry permits using only half of the sphere for the computer analysis. The elements representation is given in Fig. 9. Since the program gives stresses in cylindrical coordinates, these have been transformed to the spherical coordinates for comparison with the exact solution in Fig. 10. The accuracy of the results is noteworthy.

## C. THERMAL STRESSES IN NOZZLE

This problem concerns thermal stresses near the intersection of a cylindrical pipe and the spherical vessel. Fig. 11 gives the cross-section of the structure which is to be analysed. The material properties are:

| | | |
|---|---|---|
| Modulus of elasticity | $E = 29.3 \times 10^{6}$ | Psi |
| Poisson's ratio | $\nu = .30$ | - |
| Thermal expansion coefficient | $\alpha = 7.6 \times 10^{-6}$ | 1/F° |

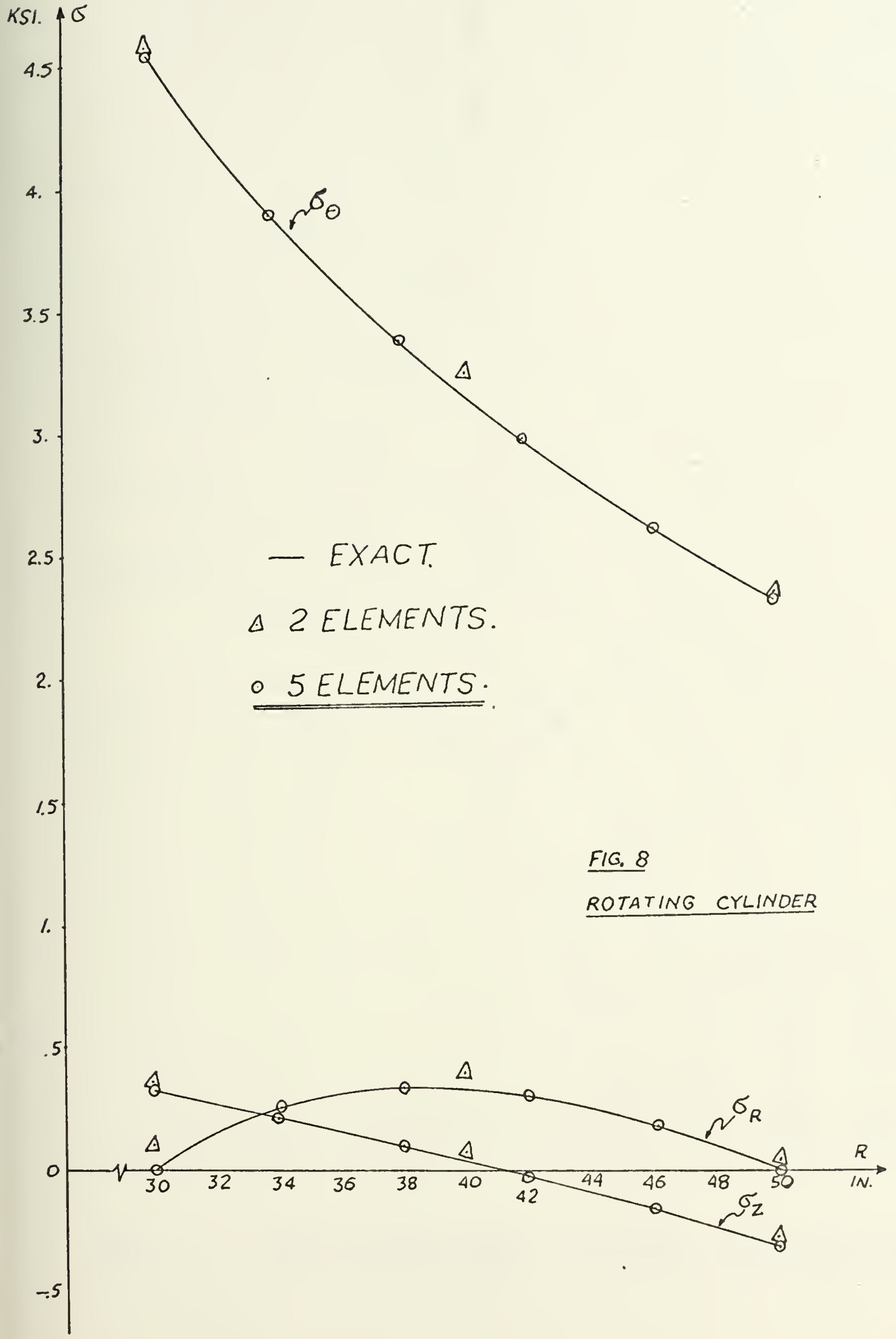


FIG. 8

ROTATING CYLINDER

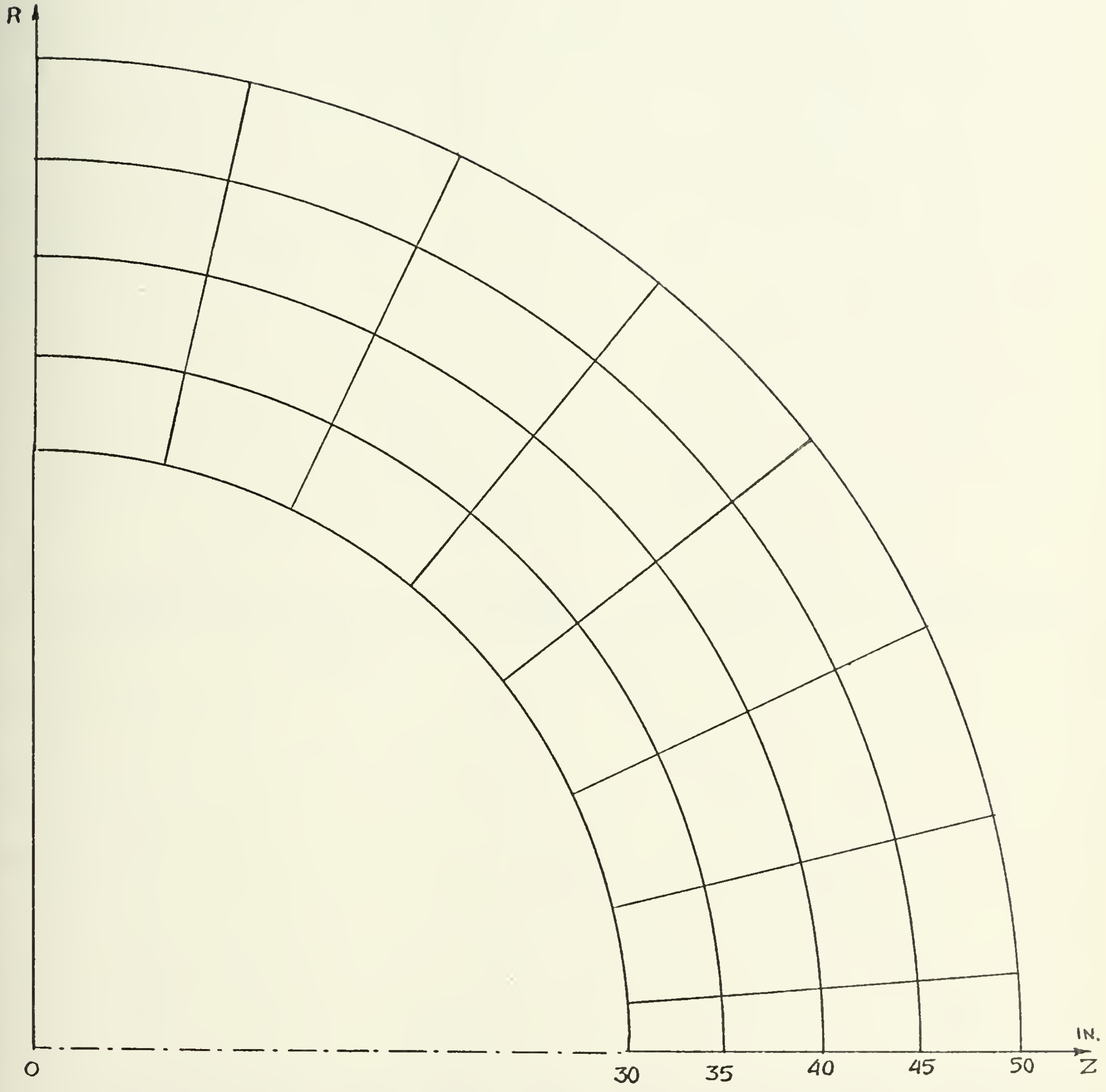


Figure 9. Hollow Semi-sphere 32 Elements Representation.

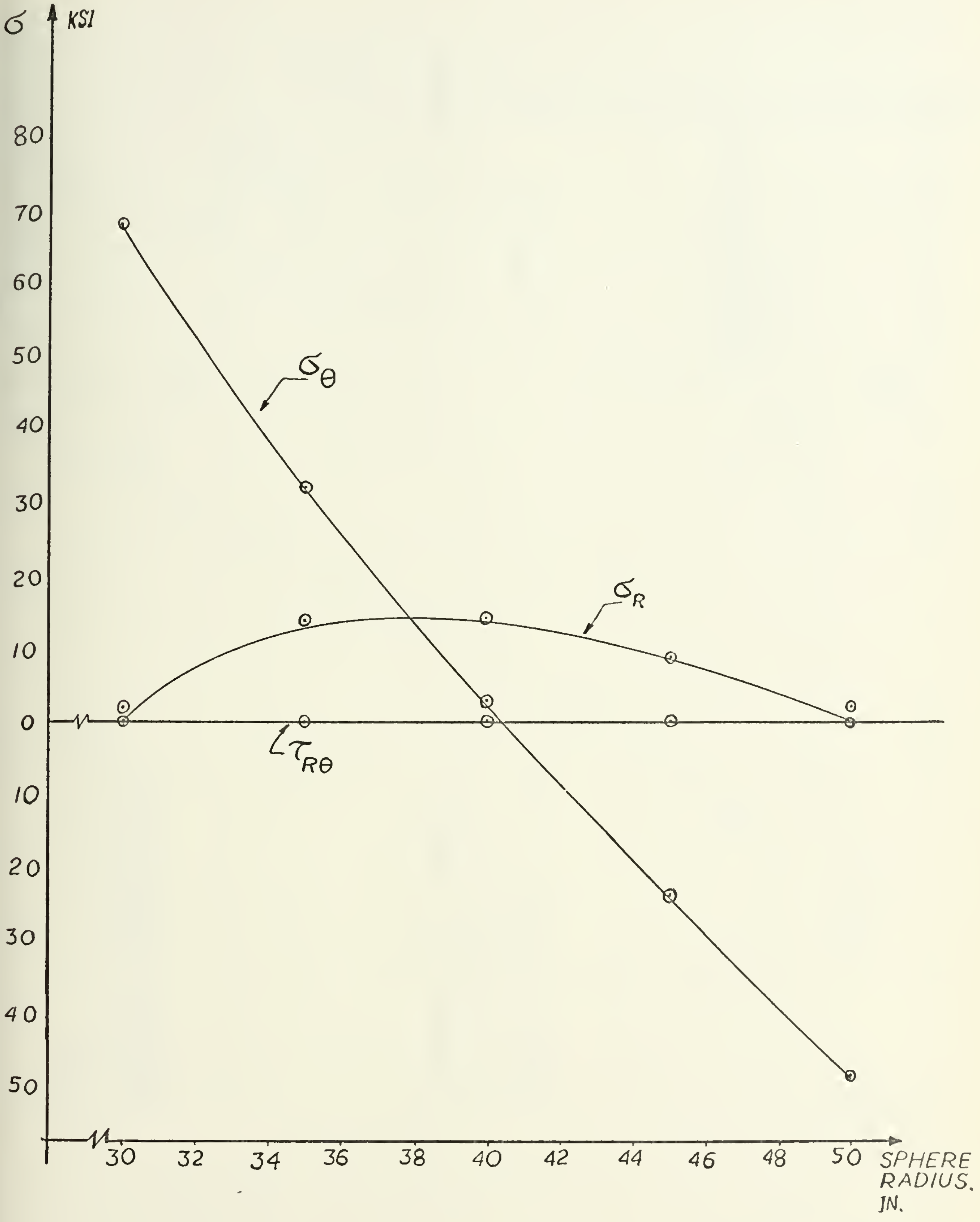
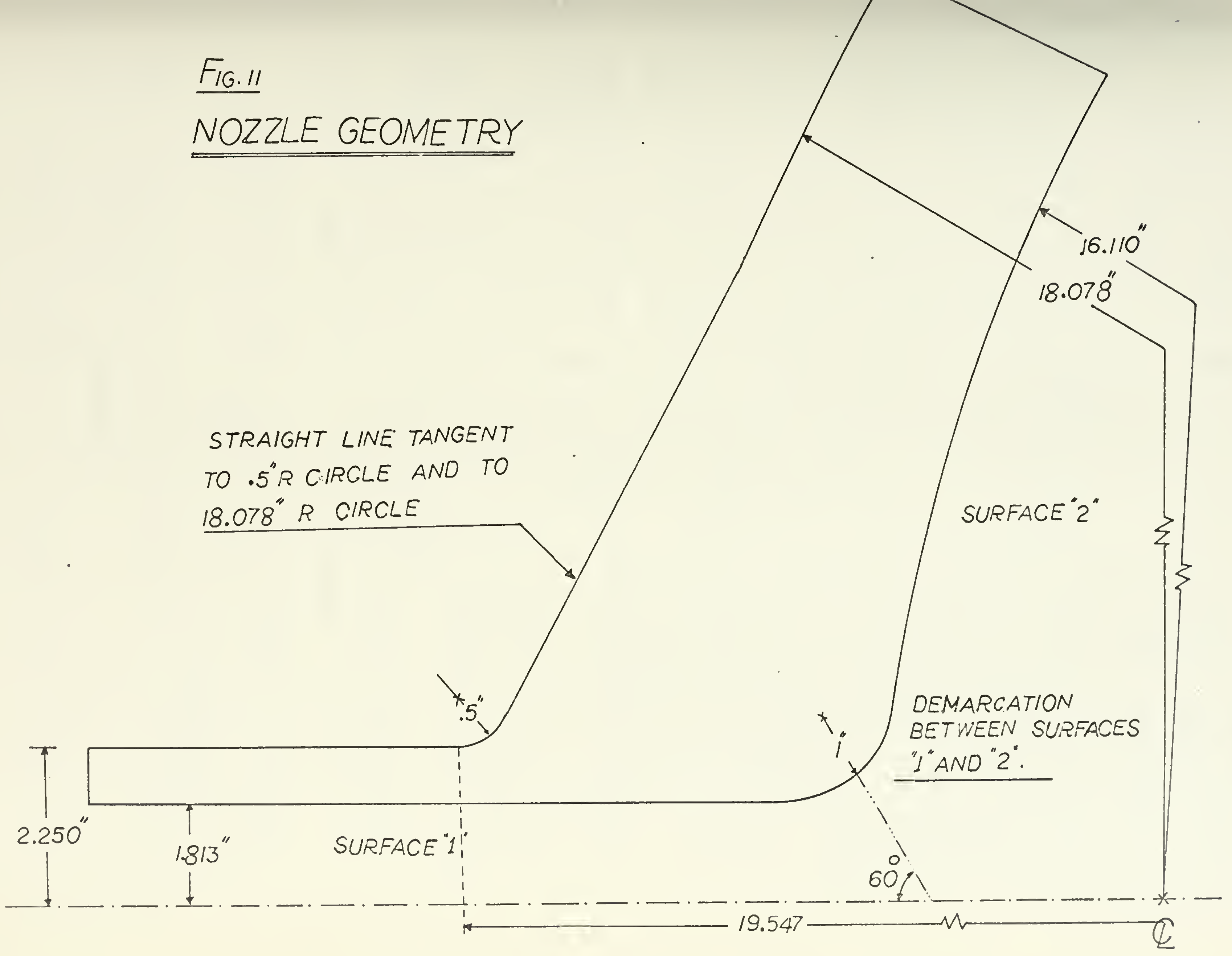


Figure 10. Thermal Stresses in Hollow Sphere T = 20 (spherical radius).

FIG. 11

NOZZLE GEOMETRY


16.110"
18.078"
STRAIGHT LINE TANGENT TO .5" R CIRCLE AND TO 18.078" R CIRCLE
SURFACE "2"
.5"
1"
DEMARCATION BETWEEN SURFACES "1" AND "2".
2.250"
1.813"
SURFACE "1"
60°
19.547
℄

| | |
|---|---|
| Thermal conductivity | $k = 10.25$ Btu/ft.hr.°F |
| Density | $\rho = 530.5$ Lbm/ft$^3$ |
| Specific heat | $c = .128$ Btu/Lbm°F |

The loading results from the thermal transient in the fluid contained in the nozzle. This fluid is in contact with the structure on surface "1" (Fig. 11) and has the entry temperature time variation as in Fig. 12. The fluid in the sphere, which is in contact with the structure on surface "2" (Fig. 11), has the constant temperature 478°F. At $\tau = 0$ structure has a uniform temperature of 478°F and is stress free. Exterior surface of the structure is insulated. The flow velocity past surface 1 is 8.5 ft/sec (inward) and there is no flow past surface 2. The surface heat transfer coefficients are 1393 and 2910 Btu/hr.ft$^2$°F for surfaces "1" and "2" respectively.

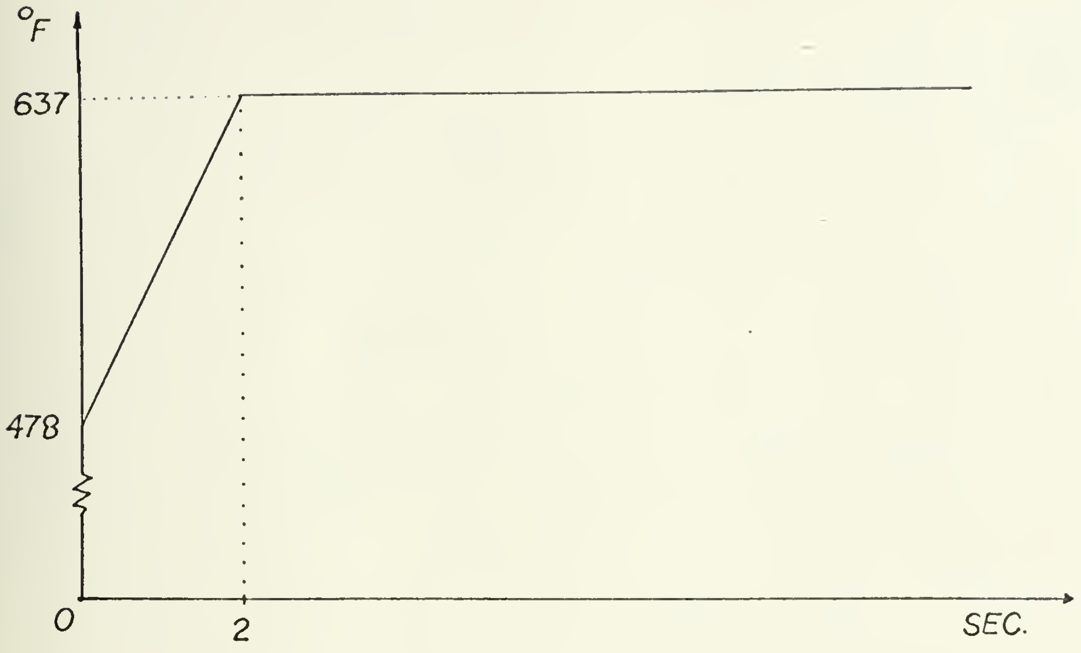
For the structural boundary condition it is assumed that the nodes on the left end cross-section of Fig. 13 are prevented from any axial displacement.

The maximum thermal stresses obtained by the program occur in element 14 of Fig. 13 as follows:

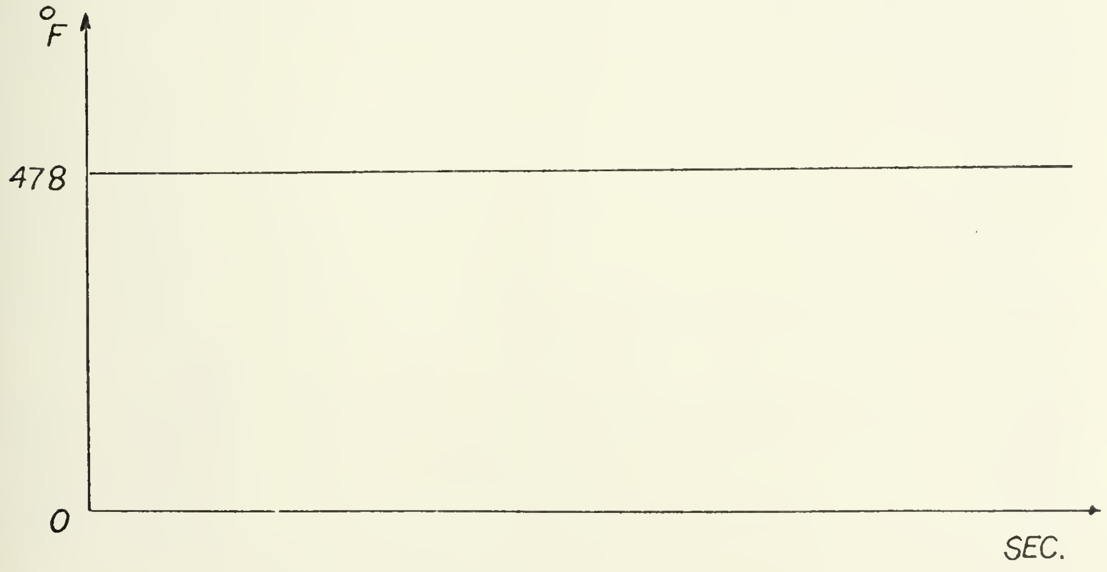
Maximum mean stress = 18.81 ksi. at time = 4 seconds;

Maximum octahedral shearing stress = 11.5 ksi. at time = 18 seconds.

These results appear to be reasonable, but no suitable comparison solution is available.



VARIATION OF ENTRY FLUID TEMPERATURE IN NOZZLE. (SURFACE "1".)



FLUID PAST SURFACE "2".

FIG. 12. Fluid Temperature - Time Histories.

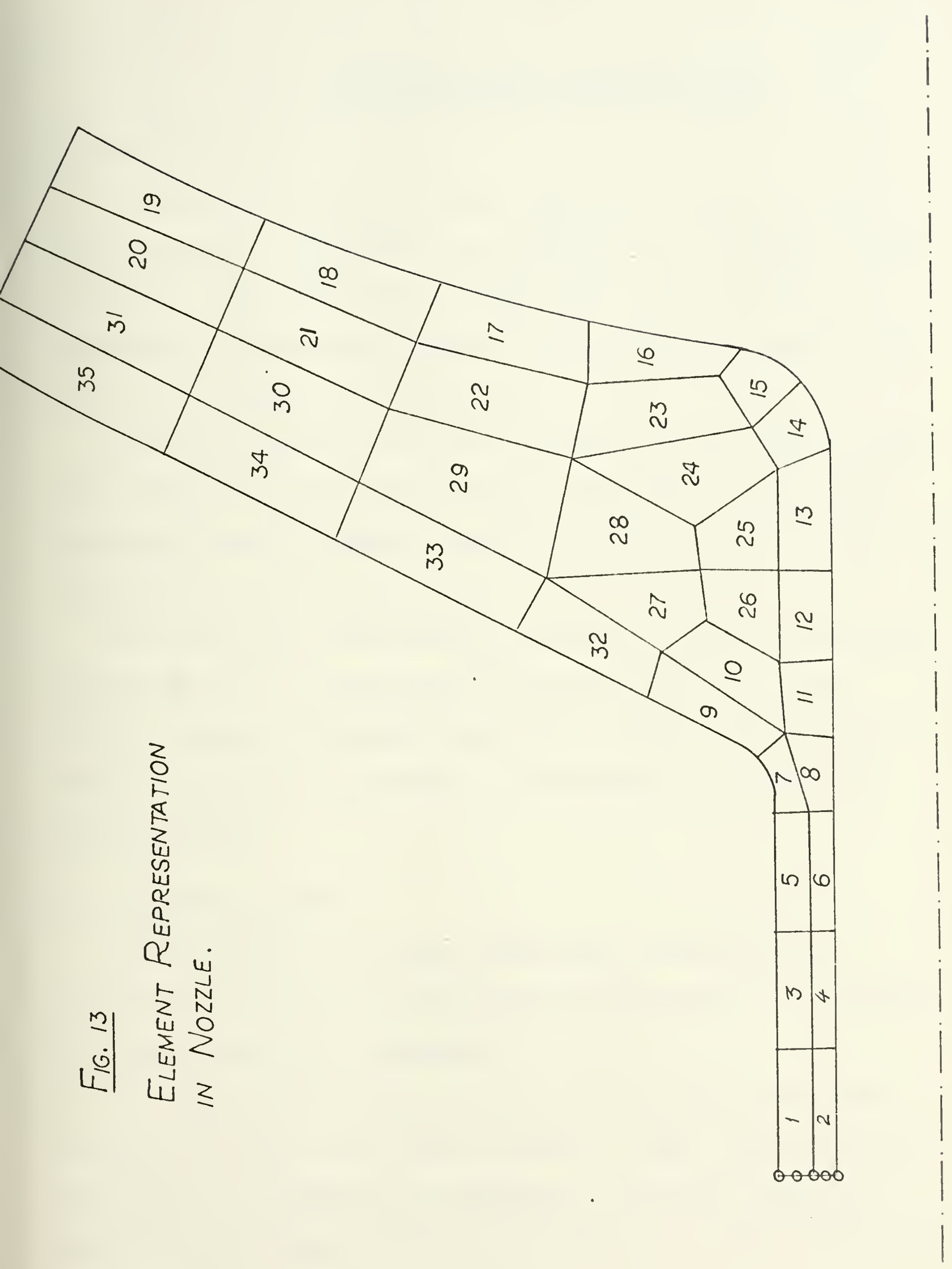


FIG. 13

ELEMENT REPRESENTATION IN NOZZLE.

# VII. CONCLUSIONS AND RECOMMENDATIONS

## A. CONCLUSIONS

A computer program has been developed for the solution of axisymmetric transient heat conduction and thermal stress problems. The system will accommodate a wide variety of geometric arrangements, thermal and structural boundary conditions, and mechanical loadings.

Although double-precision arithmetic is employed throughout, efficient algorithms for the manipulation and storage of large symmetric banded matrices allow in-core solutions with modest time requirements.

The quadratic isoparametric elements used allow accurate representation of curvilinear boundaries and the stress field. Examples presented show that a small number of elements is generally sufficient to determine stresses with good precision.

## B. RECOMMENDATIONS

Incorporation of several additional features would significantly increase the program capabilities. The following extensions are recommended.

1. Material thermal and elastic properties have been assumed constant within each element. Since such properties are generally temperature dependent, provisions should be made for periodically "updating" them during both temperature and stress solutions.

2. The surface heat transfer coefficient, taken as constant in the program, is a function of temperature and flow velocity. Provision should be made to include these effects.

3. The program presently starts every temperature solution with constant initial solid temperature. Provision for an externally specified initial temperature vector should be included.

4. The large quantity of temperature and stress results generated by the program is currently presented as digital printout. Graphical output in the form of two-dimensional contour plots of temperatures and stresses should be provided.

# APPENDIX A

## APPLICABLE FORMULAS AND EQUATIONS

PART 1

(a): Two dimensional 8-noded (parabolic) isoparametric shape functions.

Corner nodes:

$$N_i = \frac{1}{4}(1 + \xi_o)(1 + \eta_o)(\xi_o + \eta_o - 1)$$

Mid nodes:

$$\xi_i = 0, \quad N_i = \frac{1}{2}(1 - \xi^2)(1 + \eta_o)$$

$$\eta_i = 0, \quad N_i = \frac{1}{2}(1 + \xi_o)(1 - \eta^2)$$

where

$$\xi_o = \xi\, \xi_i \;, \quad \eta_o = \eta\, \eta_i$$

(b): Temperature at a point in terms of the nodal temperatures.

$$T = \sum_{i=1}^{8} N_i T_i$$

(c): Coordinates at a point in terms of the nodal coordinates.

$$R = \sum_{i=1}^{8} N_i R_i$$

$$Z = \sum_{i=1}^{8} N_i Z_i$$

(d): The Jacobian coordinate transformation matrix.

$$\underline{\underline{J}} = \begin{bmatrix} \frac{\partial Z}{\partial \xi} & \frac{\partial R}{\partial \xi} \\ \frac{\partial Z}{\partial \eta} & \frac{\partial R}{\partial \eta} \end{bmatrix}$$

(e): Element of the finite element capacitance matrix.

$$C^e_{ij} = \rho c \int N_i \, N_j \, dV$$

(f): Element of the finite element admittance matrix.

$$y^e_{ij} = k \int \overline{\nabla} N_i \cdot \overline{\nabla} N_j \, dV$$

(g): Elemental volume.

$$dV = 2\pi R \det \underline{\underline{J}} \, d\xi \, d\eta$$

PART 2

(a): One-dimensional 3-noded (parabolic) iso-parametric shape functions.

End nodes $\quad N_i = \frac{1}{2} \xi_o \, (1 + \xi_o)$

Mid node $\quad N_i = (1 - \xi^2)$

where, again, $\xi_o = \xi \, \xi_i$

(b): One-dimensional capacitance and admittance matrices.

$$\underline{\underline{C}}^e = \frac{\rho c \ell}{30} \begin{bmatrix} 4 & 2 & -1 \\ 2 & 16 & 2 \\ -1 & 2 & 4 \end{bmatrix}, \quad \underline{\underline{Y}}^{+e} = \frac{k}{3\ell} \begin{bmatrix} 7 & -8 & 1 \\ -8 & 16 & -8 \\ 1 & -8 & 7 \end{bmatrix} + \begin{bmatrix} h & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

where $\ell$ is the length of the element.

(c): The element $\underline{v}$ vector for the one-dimensional case.

$$\underline{v}^e = \langle hT_f \;\; 0 \;\; 0 \rangle^T$$

PART 3

(a): Strain-displacement relations.

$$\varepsilon_Z = \frac{\partial w}{\partial Z} = \Sigma \frac{\partial N_i}{\partial Z} w_i$$

$$\varepsilon_R = \frac{\partial u}{\partial R} = \Sigma \frac{\partial N_i}{\partial R} u_i$$

$$\varepsilon_\theta = \frac{u}{R} = \frac{\Sigma N_i U_i}{\Sigma N_i R_i}$$

$$\dot{\gamma}_{RZ} = \frac{\partial u}{\partial Z} + \frac{\partial w}{\partial R} = \Sigma \frac{\partial N_i}{\partial Z} U_i + \Sigma \frac{\partial N_i}{\partial R} w_i$$

where $N_i$ are the same isoparametric shape function as in Part 1, (a).

(b): The $\underline{\underline{B}}$ matrix.

$$\underline{\underline{B}} = \begin{bmatrix} 0 & \frac{\partial N_1}{\partial Z} & 0 & \frac{\partial V_2}{\partial Z} & \cdots\cdots & \frac{\partial N_8}{Z} \\ \frac{\partial N_1}{\partial R} & 0 & \frac{\partial N_2}{\partial Z} & 0 & \cdots\cdots & 0 \\ \frac{N_1}{R} & 0 & \frac{N_2}{R} & 0 & \cdots\cdots & 0 \\ \frac{\partial N_1}{\partial Z} & \frac{\partial N_1}{\partial R} & \frac{\partial N_2}{\partial Z} & \frac{\partial N_2}{\partial R} & \cdots\cdots & \frac{\partial N_8}{\partial R} \end{bmatrix}$$

(c): The element displacement vector.

$$\underline{\delta}^e = \langle u_1 \; w_1 \; u_2 \; w_2 \; \cdots\cdots \; u_8 \; w_8 \rangle^T$$

(d): Elasticity matrix $\underline{\underline{D}}$ for an isotropic material.

$$\underline{\underline{D}} = \frac{E(1-\nu)}{(1+\nu)(1-2\nu)} \begin{bmatrix} 1 & \frac{\nu}{1-\nu} & \frac{\nu}{1-\nu} & 0 \\ \frac{\nu}{1-\nu} & 1 & \frac{\nu}{1-\nu} & 0 \\ \frac{\nu}{1-\nu} & \frac{\nu}{1-\nu} & 1 & 0 \\ 0 & 0 & 0 & \frac{1-2\nu}{2(1-\nu)} \end{bmatrix}$$

```
C
C     ..................................................................
C     .                                                                .
C     .  A.  IDENTIFICATION                                            .
C     .                                                                .
C     .                                                                .
C     .      TITLE                     AXITTS                          .
C     .                                (AXISYMMETRIC TRANSIENT         .
C     .                                THERMAL STRESSES.)              .
C     .      COMPLETION DATE           JUNE 1972                       .
C     .      PROGRAMMER                LT. M. BAKHSHANDEHPOUR          .
C     .                                IMPERIAL IRANIAN  NAVY          .
C     .      ADVISER                   DR. ROBERT  E.  NEWTON          .
C     .                                PROF.MECH.ENG.N.P.G.S.          .
C     .                                                                .
C     .  B.  OBJECTIVE                                                 .
C     .                                                                .
C     .      THIS PROGRAM IS USED FOR THE FOLLOWING CASES:             .
C     .                                                                .
C     .      1-   TO EVALUATE TRANSIENT TEMPERATURE IN AN AXI-         .
C     .           SYMMETRIC  BODY  FOR  THE COMBINATION OF THE         .
C     .           CONSTANT TEMPERATURE, INSULATED AND  CONVEC-         .
C     .           TION THERMAL BOUNDARY CONDITION.                     .
C     .                                                                .
C     .      2-   TO EVALUATE THE STRESSES IN AN  AXISYMMETRIC         .
C     .           BODY FOR ANY COMBINATION OF AXIAL, PRESSURE,         .
C     .           CENTRIFUGAL OR ANY EXTERNALLY SPECIFIED LOAD         .
C     .           VECTOR.                                              .
C     .                                                                .
C     .      3-   TO EVALUATE TRANSIENT THERMAL STRESSES IN AN         .
C     .           AXISYMMETRIC BODY  FOR 'ANY KNOWN' TRANSIENT         .
C     .           TEMPERATURE OR USING THE RESULTS OF  STEP 1.         .
C     .           THE COMBINATION OF THERMAL LOADING WITH  ANY         .
C     .           OTHER LOADING AS IN STEP 2 MAY BE USED ALSO.         .
C     .                                                                .
C     .                                                                .
C     .  C.  METHOD OF CALCULATION                                     .
C     .                                                                .
C     .      THE FINITE ELEMENT METHOD WITH  EIGHT NODED ISO-          .
C     .      PARAMETRIC ELEMENTS IS APPLIED FOR THE ANALYSIS.          .
C     .                                                                .
C     .                                                                .
C     .  D.  USAGE                                                     .
C     .                                                                .
C     .      REFER TO THE (APPENDIX C) OF THE MECHANICAL ENG.          .
C     .      THESIS WRITTEN BY THE PROGRAMMER.                         .
C     .                                                                .
C     ..................................................................
```

```
C                                                                       00000510
C                                                                       00000520
C     ************************************************************     00000530
C                                                                       00000540
C                                                                       00000550
C     DESCRIPTION OF THE SYMBOLS.                                       00000560
C                                                                       00000570
C     AJ     IS THE JACOBIAN MATRIX OR ITS INVERSE.                     00000580
C     AL     IS THE COEFFICIENT OF THERMAL EXPANSION , ALPHA.           00000590
C     AXIALF IS THE AXIAL FORCE.                                        00000600
C     BTE    IS A DUMMY VECTOR FOR CALCULATION OF TEMPERATURE.          00000610
C     BDRYO  IS THE OUTSIDE THERMAL BOUNDARY CONDITION                  00000620
C     BDRYI  IS THE INSIDE  THERMAL BOUNDARY CONDITION.                 00000630
C     C      IS THE THERMAL CAPACITANCE MATRIX OR C+.5*DTI*Y            00000640
C     CE     IS THE ELEMENT CAPACITANCE MATRIX.                         00000650
C     COPY   IS A COPY OF PART OF STIFFNESS MATRIX.                     00000660
C     D      IS THE STANDARD D MATRIX.                                  00000670
C     DL     IS THE LOWER TRIANGLE OF D MATRIX.                         00000680
C     DNZR   IS AN ARRAY OF THE DERIVATIVES OF SHAPE FUNCTIONS          00000690
C            RESPECT TO Z AND R COORDINATES.                            00000700
C     DETA   IS THE ARRAY OF THE DERIVATIVES OF SHAPE FUNCTIONS         00000710
C            WITH RESPECT TO ETA.                                       00000720
C     DTI    IS THE TIME INCREMENT USED IN INTEGRATION.                 00000730
C     DXI    IS THE ARRAY OF THE DERIVATIVES OF THE SHAPE               00000740
C            FUNCTIONS WITH RESPECT TO XI.                              00000750
C     DENS   IS THE DENSITY.                                            00000760
C     E      IS THE YOUNG'S MODULUS OF ELASTICITY.                      00000770
C     EPSO   IS THE LOCAL INITIAL STRAIN VECTOR.                        00000780
C     ESM    IS THE ELEMENT STIFFNESS MATRIX.                           00000790
C     F      IS THE ARRAY OF VARIOUS LOAD VECTORS.                      00000800
C     FE     IS THE ELEMENT LOAD VECTOR.                                00000810
C     GC     IS THE GAUSS COEFFICIENT.                                  00000820
C     GX     IS THE GAUSS ABSCISSA.                                     00000830
C     HTCI   IS INSIDE  HEAT TRANSFER COEFFICIENT                       00000840
C     HTCO   IS OUTSIDE HEAT TRANSFER COEFFICIENT                       00000850
C     IVEC   IS THE NUMBER OF STORED TEMPERATURE VECTORS.               00000860
C     INTP   IS THE INTERVAL OF TEMPERATURE PRINT                       00000870
C     JVEC   IS THE LOCATION INDICATION OF VARIOUS LOAD                 00000880
C            VECTORS WITHIN F .                                         00000890
C     NCFIT  IS THE TOTAL NUMBER OF NODES IN CONTACT WITH               00000900
C            INSIDE FLUID.                                              00000910
C     NCFOT  IS THE TOTAL NUMBER OF NODES IN CONTACT WITH               00000920
C            OUT SIDE FLUID.                                            00000930
C     NET    IS THE NUMBER OF ELEMENTS.                                 00000940
C     NDF    IS THE NUMBER OF DEGREES OF FREEDOM.                       00000950
C     NGP    IS THE NUMBER OF GAUSS POINTS FOR INTEGRATION.             00000960
C     NNE    IS THE NUMBER OF NODES PER ELEMENT                         00000970
C     NNLT   IS THE TOTAL NUMBER OF NODES LONGITUDINALLY FIXED.         00000980
```

```
C         NNRT     IS THE TOTAL NUMBER OF NODES RADIALLY FIXED
C         NPROB    IS THE NUMBER OF PROBLEMS
C         NPASS    IS THE NUMBER OF STEPS OF TIME INTEGRATION.
C         NPNT     IS THE TOTAL NUMBER OF PRESSURE NODES.
C         NCFI     IS THE ARRAY OF NODES IN CONTACT WITH FLUID INSIDE
C         NCFO     IS THE ARRAY OF NODES IN CONTACT WITH FLUID OUTSIDE
C         OMEGA    IS THE NUMBER OF REVOLUTIONS PER MINUTE.
C         POI      IS THE POISSON'S RATIO.
C         PRES     IS THE PRESSURE.
C         R        IS THE ARRAY OF THE RADIAL COORDINATES.
C         SHP      IS THE ARRAY OF THE SHAPE FUNCTIONS.
C         SHT      IS THE SPECIFIC HEAT.
C         SSM      IS THE SYSTEM STIFFNESS MATRIX.
C         STOR     IS THE ARRAY OF STORED TEMPERATURE VECTORS.
C         SMAX     IS THE MAX. MEAN STRESS.
C         SHMAX    IS THE MAX. OCTAHEDRAL SHEAR STRESS.
C         TINIT    IS THE INITIAL TEMPERATURE OF THE BODY.
C         TK       IS THE THERMAL CONDUCTIVITY.
C         TMAXM    IS THE TIME WHEN MAX. MEAN STRESS OCCURS.
C         TMAXO    IS THE TIME WHEN MAX. OCTAHEDRAL SHEAR
C                  STRESS OCCURS.
C         TREF     IS THE REFERENCE TEMPERATURE.
C         Z        IS THE ARRAY OF THE LONGITUDINAL COORDINATES.
C
C         *****************************************************************
C
C
      IMPLICIT  REAL * 8 (A-H,O-Z)
      REAL  * 4  ENTEO,ENTEI,ENTIO,ENTII,PLOT
      COMMON /ONE/ AJ(2,2),B(4,16),COPY(37,298),D(4,4),DL(4,4),
     1DNZR(2,8),ESM(16,16),F(24,298),SSM(298,66),STOR(20,150)
C
      COMMON /TWO/ AL(5),DENS(5),DETA(8),DXI(8),E(5),EPSO(4),
     1FE(16),GC(4),GX(4),POI(5),R(149),SHP(8),SHT(5),TK(5),Z(149)
C
      COMMON /TRE/ AXIALF,DTI,EVERY,FORS1,OMEGA,HTCI,HTCO,
     1POSI,PRES,Q1,Q2,Q3,SHMAX,SMAX,TIME,TIME1,TINIT,TMAXM,TREF,
     2TMAXO,RMAXM,RMAXO,ZMAXM,ZMAXO,Q4,BIG
C
      COMMON /FOR/ NN(40,9)
C
      COMMON /FIV/ NCFI(37),NCFO(37),NNR(37),NRE(37),NNL(37),
     1NPN(37)
      COMMON /SIX/ IBAND,IEXT,INTP,IVEC,JVEC,KK,LOAD,NCFIT,NCFOT,
     1NCHECK,NET,NDF,NGP,NNE,NNLT,NNRT,NPASS,NPROB,NRPPI,NRPPO,
     3NNT,NNRE,NPNT,IPLANE,NBAND
C
      EQUIVALENCE  (C,SSM) , (Y,SSM(1,18)) , (T,SSM(1,35)) ,
```

```
     1(BTE,SSM(1,36)) , (TEMP,SSM(1,37)) , (V,SSM(1,38)) ,
     2(FT1,SSM(1,39)) , (FT2,SSM(1,40)) , (CE,SSM(1,41)) ,
     3(YE,SSM(1,42)) , (OFF1,SSM(1,43)) , (OFF2,SSM(1,44)) ,
     4(DIAG,SSM(1,45)) , (BDRYI,SSM(1,46)) , (BDRYO,SSM(1,47)) ,
     5(ENTII,SSM(1,48)) , (ENTIO,SSM(1,49)), (ENTEI,SSM(1,50)),
     6(ENTEO,SSM(1,51)),(VOL,SSM(1,52)),(VOLO,SSM(1,53))
C
      DIMENSION  C(149,33),Y(149,33),T(149),BTE(149),TEMP(149),
     1V(149),FT1(90),FT2(90),CE(8,8),YE(8,8),OFF1(37),OFF2(37),
     2DIAG(37),BDRYI(16,5),BDRYO(16,5),VOL(36),VOLO(36),ENTEO(75),
     3ENTEI(75),ENTIO(75),ENTII(75)
C
C
      CALL   INPUT
C
C        TEMPERATURE PROBLEM
C
C
      DO  2  L = 1 , NPROB
C
C
      DO  5  I = 1 , 75
      ENTEO(I) = 0.0
      ENTEI(I) = 0.0
      ENTIO(I) = 0.0
    5 ENTII(I) = 0.0
C
C
      WRITE  (6,99)  L
   99 FORMAT  (1H1,//,50X,'PROBLEM NUMBER ',I5,/,50X,
     120('*'),/,50X,20('*'),//)
      CALL   PROB
C
      IF  ((Q4.LT.0.D0).OR.(IEXT.NE.0))  GO  TO  3
C
      CALL CANDY
      DO  1  K = 1 , NPASS
      CALL   FLOW
C
      CALL   FORMV
C
      CALL   TEMPER
    1 CONTINUE
C
      IF  (Q2.GT.0.D0)  CALL  PLOT(ENTIO,ENTEO,0)
      IF  (Q3.GT.0.D0)  CALL  PLOT(ENTII,ENTEI,1)
C
      IF  (Q4.GT.0.D0)  GO  TO   2
```

```
C
C        STRESS PROBLEM
C
C
    3 CALL   STIFF
      CALL   FORMF
      IF   (OMEGA.NE.0.D0) CALL   CENTF
      IF   (PRES.NE.0.D0)  CALL   PRESS
      CALL   DISPL
      CALL   STRESS
C
C
      WRITE  (6,100)
  100 FORMAT (1H1,///,2(25X,54('*'),/),2(25X,'**',50X,'**',/),
     125X,'**',50X,'**')
      WRITE  (6,101) TMAXM , SMAX , RMAXM , ZMAXM
  101 FORMAT (25X,'**   AT TIME = ',F10.2,27X,'**',/,25X,
     1'**   MAX. MEAN STRESS IS ',1PD14.6,13X,'**',/,25X,
     2'**   AND IT OCCURS AT THE POINT WHERE',15X,'**',/,
     325X,'**   R = ',1PD14.6,8X,'Z = ',1PD14.6,3X,'**',/,
     48(25X,'**',50X,'**',/),25X,'**',50X,'**')
      WRITE  (6,102) TMAXO , SHMAX , RMAXO , ZMAXO
  102 FORMAT (25X,'**   AT TIME = ',F10.2,27X,'**',/,25X,
     1'**   MAX. OCTAHEDRAL SHEAR STRESS IS',1PD14.6,2X,
     2'**',/,25X,'**   AND IT OCCURS AT THE POINT WHERE',
     315X,'**',/,25X,'**   R= ',1PD14.6,8X,'Z = ',1PD14.6,
     44X,'**',/,4(25X,'**',50X,'**',/),2(25X,54('*'),/))
C
C
C
    2 CONTINUE
      STOP
      END
C
```

```
      SUBROUTINE  INPUT
C
C        ************************************************************
C                                                                   *
C        THIS SUBROUTINE READS AND WRITES THE INPUT DATA OF         *
C        THE GEOMETRY AND MATERIAL PROPERTIES.IT CALCULATES         *
C        THE MID NODE COORDINATES BASED ON STRAIGHT AND THE         *
C        SYSTEM OF UNITS IS SELECTED HERE.                          *
C        AT THE END THE BANDWIDTH OF THE STIFFNESS  MATRIX          *
C        IS DETERMINED.                                             *
C                                                                   *
C        ************************************************************
C
C
C
      IMPLICIT  REAL  * 8 (A-H,O-Z)
      REAL  *  4  HCO,VCO,PLOTZR
C
      COMMON /ONE/ AJ(2,2),B(4,16),COPY(37,298),D(4,4),DL(4,4),
     1DNZR(2,8),ESM(16,16),F(24,298),SSM(298,66),STOR(20,150)
C
      COMMON /TWO/ AL(5),DENS(5),DETA(8),DXI(8),E(5),EPSO(4),
     1FE(16),GC(4),GX(4),POI(5),R(149),SHP(8),SHT(5),TK(5),Z(149)
C
      COMMON /TRE/ AXIALF,DTI,EVERY,FORS1,OMEGA,HTCI,HTCO,
     1POSI,PRES,Q1,Q2,Q3,SHMAX,SMAX,TIME,TIME1,TINIT,TMAXM,TREF,
     2TMAXO,RMAXM,RMAXO,ZMAXM,ZMAXO,Q4,BIG
C
      COMMON /FOR/ NN(40,9)
C
      COMMON /FIV/ NCFI(37),NCFO(37),NNR(37),NRE(37),NNL(37),
     1NPN(37)
      COMMON /SIX/ IBAND,IEXT,INTP,IVEC,JVEC,KK,LOAD,NCFIT,NCFOT,
     1NCHECK,NET,NDF,NGP,NNE,NNLT,NNRT,NPASS,NPROB,NRPPI,NRPPO,
     3NNT,NNRE,NPNT,IPLANE,NBAND
C
      EQUIVALENCE  (C,SSM) , (Y,SSM(1,18)) , (T,SSM(1,35)) ,
     1(BTE,SSM(1,36)) , (TEMP,SSM(1,37)) , (V,SSM(1,38)) ,
     2(FT1,SSM(1,39)) , (FT2,SSM(1,40)) , (CE,SSM(1,41)) ,
     3(YE,SSM(1,42)) , (OFF1,SSM(1,43)) , (OFF2,SSM(1,44)) ,
     4(DIAG,SSM(1,45)) , (BDRYI,SSM(1,46)) , (BDRYO,SSM(1,47)) ,
     5(ENTII,SSM(1,48)) , (ENTIO,SSM(1,49)), (ENTEI,SSM(1,50)),
     6(ENTEO,SSM(1,51)),(VOL,SSM(1,52)),(VOLO,SSM(1,53))
      EQUIVALENCE  (HCO,SSM(1,54)) , (VCO,SSM(1,55))
C
      DIMENSION  C(149,33),Y(149,33),T(149),BTE(149),TEMP(149),
```

```
     1V(149),FT1(90),FT2(90),CE(8,8),YE(8,8),OFF1(37),OFF2(37),
     2DIAG(37),BDRYI(16,5),BDRYO(16,5),VOL(36),VOLO(36),ENTEO(75),
     3ENTEI(75),ENTIO(75),ENTII(75)
      DIMENSION  HCO(149) , VCO(149)
C
      DATA  BRIT/'BRIT'/
C
      READ (5,301) NET,NNT,NCN,NOFN,NMAT,NPROB
  301 FORMAT  (8I5)
      NNE = 8
C
      DO  4  I = 1 , NCN
    4 READ  (5,300)  M,R(M),Z(M)
  300 FORMAT  (I5,2F10.5)
      K=NNE + 1
      READ (5,302)   ((NN(I,J),J=1,K),I=1,NET)
  302 FORMAT (9I5)
C
C
C     CALCULATION OF MID-NODE COORDINATES.
C
C
      DO  5  L = 1 , NET
      R(NN(L,2)) = (R(NN(L,1))+R(NN(L,3))) / 2.D0
      Z(NN(L,2)) = (Z(NN(L,1))+Z(NN(L,3))) / 2.D0
      R(NN(L,4)) = (R(NN(L,3))+R(NN(L,5))) / 2.D0
      Z(NN(L,4)) = (Z(NN(L,3))+Z(NN(L,5))) / 2.D0
      R(NN(L,6)) = (R(NN(L,5))+R(NN(L,7))) / 2.D0
      Z(NN(L,6)) = (Z(NN(L,5))+Z(NN(L,7))) / 2.D0
      R(NN(L,8)) = (R(NN(L,7))+R(NN(L,1))) / 2.D0
    5 Z(NN(L,8)) = (Z(NN(L,7))+Z(NN(L,1))) / 2.D0
C
C
      IF  (NOFN. EQ . 0)  GO  TO  7
      DO  6  I = 1 , NOFN
    6 READ  (5,300)  M,R(M),Z(M)
C
    7 CONTINUE
C
      DO  19  I = 1 , NNT
      HCO(I) = Z(I)
      VCO(I) = R(I)
   19 CONTINUE
C
C
      CALL  PLOTZR(HCO,VCO,NNT)
C
C
```

```
      DO  1  I = 1 , NMAT
    1 READ  (5,303) E(I),AL(I),POI(I),TK(I),DENS(I),SHT(I)
  303 FORMAT  (2D12.4,4F8.3)
C
C
C     ARE THE UNITS IN METRIC OR BRITISH UNIT SYSTEM?
C
      READ (5,310) Q1
  310 FORMAT  (A4)
C
      WRITE  (6,200)
  200 FORMAT (1H1,45X,'** TRANSIENT THERMAL STRESS PROBLEM',
     1' **',/,49X,32('_'),//)
C
C
      WRITE  (6,201)  NET,NNT,NCN,NMAT,NPROB
  201 FORMAT  (/,15X,'TOTAL NUMBER OF ELEMENTS.....=',I5,
     1/,15X,'TOTAL NUMBER OF NODES........=',I5,/,15X,
     2'TOTAL NUMBER OF CORNER NODES.=',I5,/,15X,'TOTAL ',
     3'NUMBER OF MATERIALS....=',I5,/,15X,'TOTAL NUMBER ',
     4'OF PROBLEMS.....=',I5,//)
C
C
      WRITE  (6,204)
  204 FORMAT (//,15X,'ELEMENT NO.',8X,'GLOBAL NODE NUMBER',
     132X,'MATERIAL NO.',/)
C
C
      DO  2  I=1 , NET
    2 WRITE  (6,205)  I,(NN(I,J),J=1,K)
  205 FORMAT (15X,I5,10X,9(I5,2X))
C
C
      IF  (Q1.NE.BRIT)  GO  TO  10
C
      WRITE (6,211)
      WRITE  (6,213)
      GO TO 11
C
C
   10 WRITE  (6,212)
      WRITE  (6,215)
   11 WRITE (6,202)
C
C
  202 FORMAT(/,15X,'THE RADIAL AND LONGITUDINAL',
     1' COORDINATES',///,15X,'NODE NO.',14X,'**R**',
     212X,'**Z**',/)
```

```
C
C
      WRITE  (6,216) (I,R(I),Z(I),I=1,NNT)
  216 FORMAT  (16X,I4,13X,F10.4,6X,F10.4)
C
  211 FORMAT (//,15X,'** THE FOLLOWING UNITS ARE USED **',/,
     115X,'NAME',16X,'SYMBOL',14X,'UNIT',/,15X,4('*'),16X,
     2'******',14X,'****',///,15X,'TIME',16X,'TIME,DII',
     312X,'SECOND',/,15X,'TEMPERATURE',9X,'T,TEMP,FT',11X,
     4'DEGREE FAHRENHEIT',/,15X,'VELOCITY',12X,'VEL',17X,
     5'FEET PER SECOND',/,15X,'COORDINATES',9X,'R,Z',
     617X,'INCHES',/,15X,'MOD. OF ELASTICITY',2X,'E',19X,
     7'POUND FORCE/SQ.INCHES',/,15X,'POISSON S RATIO',5X,
     8'POI',17X,'DIMENSIONLESS')
C
C
  213 FORMAT  (15X,'COEF.OF THER.EXP.',15X,'THER.CONDUCTIVITY',
     13X,'AL',18X,'1/DEGREE F.',/,15X,'THER.CONDUCTIVITY',
     23X,'TK',18X,'BTU/FT.HR.,DEG.F',/,15X,'DENSITY',13X,
     3'DENS',16X,'LB/FT.CUBE',/,15X,'SPECIFIC HEAT',7X,
     4'SHT',17X,'BTU/LB.DEG.F',/,15X,'HEAT TRANSFER COEF',
     52X,'HTCO,HTCI',11X,'BTU/FT.SQ,HR.,DEG.F',/)
C
C
  212 FORMAT (//,15X,'THE FOLLOWING UNITS ARE BEING USED',
     1//,15X,'NAME',16X,'SYMBOL',14X,'UNIT',/,15X,'****',
     216X,'******',14X,'****',///,15X,'TIME',16X,'TIME,DTI',
     312X,'SECOND',/,15X,'TEMPERATURE',9X,'T,TEMP,FT',11X,
     4'CELSIUS     ',/,15X,'VELOCITY',12X,'VEL',17X,'METER/',
     5'SEC',/,15X,'COORDINATES',9X,'R,Z',17X,'CENTIMETER',
     6/,15X,'MOD.OF ELASTICITY',3X,'E',19X,'KG./CM.SQ',/,
     715X,'COEF.OF THER.EXP.',3X,'AL',18X,'1/CELSIUS')
C
C
  215 FORMAT  (15X,
     1     'THER.CONDUCTIVITY',3X,'TK',18X,'CAL/HR.,M.,CEL',
     2'SIUS    ',/,15X,'DENSITY',13X,'DENS',16X,'GM./CM.C',
     3'UBE',/,15X,'SPECIFIC HEAT',7X,'SHT',17X,'CAL./KG.',
     4'CELSIUS    ',/,15X,'HEAT TRANSFER COEF',2X,'HTCO,H',
     5'TCI',11X,'CAL/HR,M.SQ,CELSIUS',/)
C
      WRITE  (6,206)
  206 FORMAT (//,15X,'* PROPERTIES OF THE MATERIALS *',//,
     115X,'MATERIAL',5X,'MODULUS OF',5X,'POISSON S',5X,
     2'THERM.EXP,',5X,'THERMAL',10X,'VOLUME',6X,'SPECIFIC')
C
C
      WRITE  (6,207)
```

```
  207 FORMAT (15X,'NUMBER',7X,'ELASTICITY',5X,'RATIO',9X,
     1'COEFFICIENT',4X,'CONDUCTIVITY',5X,'DENSITY',5X,
     2'HEAT',//)
C
C
      DO  3  I=1,NMAT
    3 WRITE  (6,209) I,E(I),POI(I),AL(I),TK(I),DENS(I),SHT(I)
  209 FORMAT (14X,I5,6X,F10.1,6X,F9.5,5X,F10.7,6X,F8.4,9X,
     1F8.4,5X,F8.4,/)
C
C
C          CONVERSION OF THE SYSTEM OF UNITS.
C
      IF  (Q1.EQ.BRIT)  GO  TO  60
      TREF = 20.D0
      DO  61  I = 1 , NMAT
      DENS(I) = DENS(I) / 1000.D0
   61 E(I) = E(I) * 100.D0
      GO  TO  63
   60 TREF = 68.D0
      DO  62  I = 1 , NMAT
      DENS(I) = DENS(I) / 1728.D0
   62 TK(I) = TK(I) / 14200.D0
   63 CONTINUE
C
C
C             DETERMINATION  OF  THE  HALF-BAND-WIDTH  OF THE
C             SYSTEM  STIFFNESS MATRIX
C
C
      IDIF = 0
      NNE1 = NNE - 1
      DO  22  I = 1 , NET
      IDIF1 = 0
C
C
      DO  21  J = 2 , NNE1
      IDIF2 = NN(I,1) - NN(I,J)
      IF  (IDIF2. LT . 0 )  IDIF2 = -IDIF2
      IF  (IDIF2.GT.IDIF1)  IDIF1 = IDIF2
   21 CONTINUE
C
      IF  (IDIF1.GT.IDIF)   IDIF = IDIF1
C
   22 CONTINUE
C
      NBAND = IDIF + 1
      IBAND = 2 * NBAND
```

```
      NDF = 2 * NNT
C
      WRITE  (6,220)  IBAND , NDF
  220 FORMAT (/,15X,'HALF-BAND-WIDTH OF THE STIFFNESS MATRIX IS',
     1I8,/,15X,'NUMBER OF DEGREES OF FREEDUM OF THE SYSTEM IS',I5,/)
C
      RETURN
      END
C
C
```

```
      SUBROUTINE  PROB
C
C
C     ************************************************************
C                                                                *
C     FOR EACH PROBLEM , IN THIS SUBROUTINE THE THERMAL          *
C     INITIAL AND BOUNDARY CONDITIONS AND  THE  VARIOUS          *
C     OTHER TYPES OF LOADING WITH SPECIFIED  STRUCTURAL          *
C     BOUNDARY CONDITIONS ARE BEING READ IN AND PRINTED          *
C     OUT. THE STEP SIZE OF TIME INTEGRATION , INTERVAL          *
C     OF TEMPERATURE PRINTS IF REQUIRED , AND THE NUMBER         *
C     OF THERMAL LOAD VECTORS WANTED ARE READ IN HERE.           *
C                                                                *
C     ************************************************************
C
C
      IMPLICIT  REAL * 8 (A-H,O-Z)
C
      COMMON /ONE/ AJ(2,2),B(4,16),COPY(37,298),D(4,4),DL(4,4),
     1DNZR(2,8),ESM(16,16),F(24,298),SSM(298,66),STOR(20,150)
C
      COMMON /TWO/ AL(5),DENS(5),DETA(8),DXI(8),E(5),EPSO(4),
     1FE(16),GC(4),GX(4),POI(5),R(149),SHP(8),SHT(5),TK(5),Z(149)
C
      COMMON /TRE/ AXIALF,DTI,EVERY,FORS1,OMEGA,HTCI,HTCO,
     1POSI,PRES,Q1,Q2,Q3,SHMAX,SMAX,TIME,TIME1,TINIT,TMAXM,TREF,
     2TMAXO,RMAXM,RMAXO,ZMAXM,ZMAXO,Q4,BIG
C
      COMMON /FOR/ NN(40,9)
C
      COMMON /FIV/ NCFI(37),NCFO(37),NNR(37),NRE(37),NNL(37),
     1NPN(37)
      COMMON /SIX/ IBAND,IEXT,INTP,IVEC,JVEC,KK,LOAD,NCFIT,NCFOT,
     1NCHECK,NET,NDF,NGP,NNE,NNLT,NNRT,NPASS,NPROB,NRPPI,NRPPO,
     3NNT,NNRE,NPNT,IPLANE,NBAND
C
      EQUIVALENCE  (C,SSM) , (Y,SSM(1,18)) , (T,SSM(1,35)) ,
     1(BTE,SSM(1,36)) , (TEMP,SSM(1,37)) , (V,SSM(1,38)) ,
     2(FT1,SSM(1,39)) , (FT2,SSM(1,40)) , (CE,SSM(1,41)) ,
     3(YE,SSM(1,42)) , (OFF1,SSM(1,43)) , (OFF2,SSM(1,44)) ,
     4(DIAG,SSM(1,45)) , (BDRYI,SSM(1,46)) , (BDRYO,SSM(1,47)) ,
     5(ENTII,SSM(1,48)) , (ENTIO,SSM(1,49)), (ENTEI,SSM(1,50)),
     6(ENTEO,SSM(1,51)),(VOL,SSM(1,52)),(VOLO,SSM(1,53))
C
      DIMENSION  C(149,33),Y(149,33),T(149),BTE(149),TEMP(149),
     1V(149),FT1(90),FT2(90),CE(8,8),YE(8,8),OFF1(37),OFF2(37),
```

```
     2DIAG(37),BDRYI(16,5),BDRYO(16,5),VOL(36),VOLO(36),ENTEO(75),
     3ENTEI(75),ENTIO(75),ENTII(75)
C
      DATA  BRIT/'BRIT'/
C
      TIME = 0.D0
      JVEC = 0
      SMAX = 0.D0
      SHMAX = 0.D0
C
      NGP = 4
      GX(1) = -.861136311594053
      GX(2) = -GX(1)
      GX(3) = .339981043584856
      GX(4) = -GX(3)
      GC(1) = .347854845137454
      GC(2) = GC(1)
      GC(3) = .652145154862546
      GC(4) = GC(3)
      PAI = 3.141592653589793
C
C
C         NNRE IS THE TOTAL NUMBER OF NODES AT RIGHT HAND END.
C         NRE(I) IS THE ARRAY OF GLOBAL NODE NUMBER FOR THE
C         RIGHT HAND END.
C         IPLANE.GT.0  IF PLANES REMAIN PLANE AT THE ENDS.
C         IPLANE.EQ.0  IF ONE END IS FREE OR RADIALLY FIXED.
C
C         IEXT  IS AN INDICATION IF TEMPERATURE VECTORS ARE
C         GENERATED OR BEING READ IN AS FOLLOWES.
C         IEXT.EQ.0   TEMPERATURE VECTORS ARE GENERATED.
C         IEXT.GT.0   TEMPERATURE VECTORS ARE INPUTED.
C
C
C
      READ (5,101)OMEGA,PRES,IEXT,LOAD,NNLT,NNRT,IPLANE
C
      IF  (OMEGA.EQ.0.D0)  GO  TO  2
      JVEC = JVEC + 1
      WRITE  (6,203)  OMEGA
    2 N = 0
      IF  (PRES.EQ.0.D0)  GO  TO  6
      READ  (5,102)  NPNT,(NPN(I),I=1,NPNT)
C
C         NPNT  IS THE TOTAL NUMBER OF PRESSURE NODES
C         NPN(I)  IS THE GLOBAL NODE NUMBER OF THE PRESSURE NODE I .
C
      WRITE  (6,444)  NPNT,(NPN(I),I=1,NPNT)
```

```
  444 FORMAT  (15X,'TOTAL NUMBER OF PRESSURE NODES IS ',I5,' ',
     1'AND THEY ARE AS :',/,3(15X,20(I3,' ,'),/),/)
      JVEC = JVEC + 1
      WRITE  (6,207)  PRES
    6 CONTINUE
C
      IF  (LOAD.EQ.0)  GO  TO  50
      JVEC = JVEC + 1
      L = IVEC + JVEC
      READ  (5,108)  (F(L,I) , I=1 , NDF)
      WRITE  (6,333) (F(L,I) , I=1 , NDF)
  333 FORMAT  (/,15X,'THE EXTERNAL LOAD VECTOR IN ORDER OF ',
     1'THE NODE NUMBERS IS :',/,5(15X,F10.4),/)
   50 CONTINUE
C
      IF (NNLT.NE.0) READ (5,304) (NNL(I),I=1,NNLT)
      IF (NNLT.NE.0) WRITE (6,308) NNLT,(NNL(I),I=1,NNLT)
  308 FORMAT (/,15X,'TOTAL NUMBER OF NODES FIXED IN THE ',
     1'LONGITUDINAL DIRECTION IS ',I5,' AND THEY ARE AS:',
     2/,3(15X,20(I3,' ,'),/),/)
C
      IF (NNRT.NE.0) READ (5,304) (NNR(I),I=1,NNRT)
      IF (NNRT.NE.0) WRITE (6,309) NNRT,(NNR(I),I=1,NNRT)
  309 FORMAT (/,15X,'TOTAL NUMBER OF NODES FIXED IN THE ',
     1'RADIAL DIRECTION ARE',I5,' AND THEY ARE AS :',/,
     23(15X,20(I3,' ,'),/),/)
  304 FORMAT  (10I5)
C
      IF (IPLANE.NE.0)  READ (5,102) NNRE,(NRE(I),I=1,NNRE)
      IF (IPLANE.NE.0) WRITE (6,214) NNRE,(NRE(I),I=1,NNRE)
  214 FORMAT (/,15X,'TOTAL NUMBER OF NODES FIXED IN LONGIT',
     1'UDINAL DIRECTION OF RIGHT HAND END IS ',I5,' AND TH',
     3'EY ARE AS:',/,2(15X,20(I3,' ,'),/),/)
C
      READ  (5,100)  TINIT , Q2 , Q3 , Q4 , AXIALF
C
      IF  (AXIALF.EQ.0.D0)  WRITE  (6,220)
  220 FORMAT  (/,15X,'THERE IS NO AXIAL FORCE ON THE BODY',/)
      IF  (AXIALF.NE.0.D0)  WRITE  (6,221)  AXIALF
  221 FORMAT  (/,15X,'AXIAL FORCE = ',F10.4,/)
C
C
      IF  (IEXT.NE.0)  GO  TO  52
      IF  (Q4.LT.0.D0)  GO  TO  57
C
C
C        IF  CONVECTION   BDRY.  OUTSIDE   Q2 > 0.D0
C        IF  INSULATED    BDRY.  OUTSIDE   Q2 = 0.D0
```

```
C
C
C        IF  CONVECTION   BDRY.   INSIDE   Q3 > 0.D0
C        IF  INSULATED    BDRY.   INSIDE   Q3 = 0.D0
C
C
C        Q4.GT.0.D0  TEMPERATURE PROBLEM IS TO BE SOLVED ONLY
C        Q4 = 0.D0   STRESS PROBLEM IS TO BE SOLVED ALSO.
C        Q4.LT.0.D0  STRESS PROBLEM IS TO BE SOLVED ONLY.
C
C
      IF  (Q2. GT . 0.D0)  GO  TO  1
      GO  TO  3
    1 CONTINUE
C
      READ  (5,103)  HTCO , TEMPO , NCFOT , NRAMPO
      N = NCFOT
      READ  (5,102)  (NCFO(I), I = 1 , NCFOT)
      NRPPO = NRAMPO + 1
C
C
      READ  (5,104)  ((BDRYO(I,J),J=1,3),BDRYO(I,5),I = 2,NRPPO)
C
C     BDRYO(I,1) IS THE INITIAL TIME, TO BE READ IN FOR
C     EACH RAMP SEGMENT.
C     BDRYO(I,2) IS THE INITIAL TEMPERATURE , TO BE READ IN.
C     BDRYO(I,3) IS THE FINAL TEMPERATURE TO BE READ IN IF IT DIFFERS
C     FROM THE NEXT INITIAL TEMPERATURE,
C     BDRYO(I,4) IS THE SLOPE , WHICH IS CALCULATED BY THE
C     PROGRAM.
C     BDRYO(I,5) IS THE VELOCITY OF THE FLUID TO BE READ IN
C     IF  IT DIFFERS FROM THE PREVIOUS RAMP VELOCITY
C
C
      WRITE  (6,2021) HTCO
C
C
      DO  4  I = 1 , NCFOT
      FT2(I) = TEMPO
    4 FT1(I) = TEMPO
C
C
      WRITE  (6,201)  (NCFO(I),FT1(I),I=1,NCFOT)
C
C
      BDRYO(1,1) = 0.D0
      BDRYO(1,2) = TEMPO
      BDRYO(1,4) = 0.D0
```

```
      BDRYO(1,5) = 0.D0
      BDRYO(NRPPO,4) = 0.D0
      BDRYO(NRPPO+1,1) = 999999.D0
C
      DO  5  I = 2 , NRAMPO
      IF  (BDRYO(I,3).EQ.0.D0)  BDRYO(I,3) = BDRYO(I+1,2)
      IF  (BDRYO(I,5).EQ.0.D0)  BDRYO(I,5) = BDRYO(I-1,5)
      DUM = BDRYO(I,3) - BDRYO(I,2)
    5 BDRYO(I,4) = DUM/(BDRYO(I+1,1) - BDRYO(I,1))
C
C
      WRITE  (6,202)  (BDRYO(I,1),BDRYO(I,2),BDRYO(I+1,1),
     1BDRYO(I,3),BDRYO(I,4),BDRYO(I,5),I = 2,NRPPO)
C
      IF  (Q1.EQ.BRIT)  GO  TO  20
      DO  21  I = 2 , NRPPO
   21 BDRYO(I,5) = BDRYO(I,5) * 100.D0
      GO  TO  18
   20 DO  22  I = 2 , NRPPO
   22 BDRYO(I,5) = BDRYO(I,5) * 12.D0
C
      GO  TO  18
C
C
    3 WRITE  (6,1000)
   18 CONTINUE
C
      IF  (Q3. GT .0D0)  GO  TO  7
      WRITE  (6,1001)
      GO  TO  9
C
    7 READ  (5,103)  HTCI , TEMPI , NCFIT , NRAMPI
      READ  (5,102)  (NCFI(I) , I = 1 , NCFIT)
C
      NRPPI = NRAMPI + 1
      READ  (5,104)  ((BDRYI(I,J),J=1,3),BDRYI(I,5),I=2,NRPPI)
C
C
      WRITE  (6,2022)  HTCI
C
C
      DO  10  I = 1 , NCFIT
      K = N + I
      FT1(K) = TEMPI
   10 FT2(K) = TEMPI
C
C
      DO  19  I = 1 , NCFIT
```

```
      K = N + I
   19 WRITE  (6,201)  NCFI(I) , FT1(K)
C
C
      BDRYI(1,1) = 0.D0
      BDRYI(1,2) = TEMPI
      BDRYI(1,4) = 0.D0
      BDRYI(1,5) = 0.D0
      BDRYI(NRPPI,4) = 0.D0
      BDRYI(NRPPI+1,1) = 999999.D0
C
C
C
      DO  11  I = 2 , NRAMPI
      IF  (BDRYI(I,3).EQ.0.D0)  BDRYI(I,3) = BDRYI(I+1,2)
      IF  (BDRYI(I,5).EQ.0.D0)  BDRYI(I,5) = BDRYI(I-1,5)
      DUM = BDRYI(I,3) - BDRYI(I,2)
   11 BDRYI(I,4) = DUM / (BDRYI(I+1,1)-BDRYI(I,1))
C
      WRITE  (6,202)  (BDRYI(I,1),BDRYI(I,2),BDRYI(I+1,1),
     1BDRYI(I,3),BDRYI(I,4),BDRYI(I,5),I=2,NRPPI)
C
      IF  (Q1.EQ.BRIT)  GO  TO  23
      DO  24  I = 2 , NRPPI
   24 BDRYI(I,5) = BDRYI(I,5) * 100.D0
      GO  TO  9
   23 DO  25  I = 2 , NRPPI
   25 BDRYI(I,5) = BDRYI(I,5) * 12.D0
C
    9 READ  (5,106)  DTI , TIME1 , EVERY , IVEC , INTP
C
C
      TS = TIME1 + EVERY * (IVEC-1)
C
      WRITE  (6,215)  DTI
  215 FORMAT  (15X,'STEP SIZE OF TIME INTEGRATION IS ',F10.4,/)
C
      DO  14  I = 1,NNT
   14 TEMP(I) = TINIT
      WRITE  (6,204)  TINIT
      WRITE  (6,205)  TREF
      WRITE  (6,206)  IVEC
C
      WRITE  (6,208)  TIME1
      WRITE  (6,209)  EVERY
      IF  (INTP.EQ.0)  GO  TO  40
      WRITE  (6,210)  INTP
      GO  TO  41
```

```
   40 WRITE  (6,211)
   41 CONTINUE
C
  208 FORMAT  (15X,'THE FIRST TEMP. VECTOR IS KEPT AT TIME ',
     1F10.4,/)
C
  209 FORMAT  (15X,'THEN TEMP. VECTORS ARE KEPT AT EVERY',
     13X,F10.4,'   SECONDS',/)
C
  210 FORMAT  (15X,'TEMPERATURES ARE PRINTED FOR EVERY',5X,I5,
     1'  INTERVAL OF TIME',//)
C
  211 FORMAT  (15X,'TEMPERATURES ARE NOT PRINTED',/)
C
C
C
      KK = 1
      NPASS = TS / DTI
      NPASS = NPASS + NPASS/10
C
C
  201 FORMAT (15X,'NODE',I5,2X,'HAS CONTACT WITH FLUID AT',2X,F10.4,
     13X,'DEGREE TEMPERATURE')
C
  202 FORMAT (//,15X,'ENTRY TEMPERATURE AND TIME RELATION',
     1/,15X,34('*'),//,15X,'INITIAL TIME',5X,'INITIAL ',
     2'TEMPERATURE',10X,'FINAL TIME',5X,'FINAL TEMPERATURE',
     310X,'SLOPE',5X,'VELOCITY',//,(17X,F10.4,10X,F10.4,
     414X,F10.2,8X,F10.4,10X,F10.4,5X,F10.4))
C
C
  203 FORMAT  (/,15X,'THE SYSTEM IS ROTATING ',F10.5,' REV',
     1'OLUTIONS PER MINUTE',/)
C
C
  204 FORMAT (//,15X,'INITIAL TEMPERATURE OF THE SOLID IS',
     1'  ',F10.3,/)
  205 FORMAT  (15X,'REFERENCE  TEMPERATURE IS',12X,F10.3,/)
  206 FORMAT  (15X,'NUMBER OF STORED TEMPERATURE VECTORS IS',I6,/)
C
  207 FORMAT  (15X,'INSIDE PRESSURE IS',19X,F10.5,/)
C
C
 1000 FORMAT (//,15X,'OUTSIDE BDRY. COND. IS INSULATED',/)
 1001 FORMAT (//,15X,'INSIDE  BDRY. COND. IS INSULATED',/)
C
C
 2021 FORMAT (//,15X,'OUTSIDE BDRY. COND. IS CONVECTION',//,
```

```
     115X,'AND THE HEAT TRANSFER COEF. IS ',1PD14.6,//)
 2022 FORMAT (//,15X,'INSIDE  BDRY. COND. IS CONVECTION',//,
     115X,'AND THE HEAT TRANSFER COEF. IS ',1PD14.6,//)
C
C
   52 IF  (IEXT.EQ.0)  GO  TO  54
      NNT1 = NNT + 1
      READ  (5,109)  ((STOR(I,J),J=1,NNT1),I=1,IEXT)
      IVEC = IEXT
      WRITE  (6,212)  IEXT
  212 FORMAT  (/,15X,I5,' TEMPERATURE VECTORS ARE READ IN AND',
     1/,15X,'     TEMPERATURES ARE IN ORDER  OF GLOBAL NODE ',
     2'NUMBERS :',/)
C
      DO  56  I = 1 , IEXT
      DO  56  J = 1 , NNT
   56 WRITE  (6,213)  J , STOR(I,J)
  213 FORMAT    (15X,'NODE(',I3,')  ',F10.3)
   54 CONTINUE
      IF  (Q1.EQ.BRIT)  GO  TO  60
C
      GO  TO  62
C
   60 IF  (Q2.GT.0.DO)  HTCO = HTCO / 518400.DO
      IF  (Q3.GT.0.DO)  HTCI = HTCI / 518400.DO
C
      GO  TO  58
C
   57 IVEC = 1
      STOR(1,NNT+1) = 0.DO
      DO  59  J = 1 , NNT
   59 STOR(1,J) = TREF
   58 CONTINUE
   62 OMEGA = OMEGA * (PAI / 30.DO)
C
      BIG = 10.DO**20
  100 FORMAT  (5F10.4)
  101 FORMAT (2F10.4,5I5)
C
  102 FORMAT  (12I5)
  103 FORMAT  (D16.4,F10.4,2I5)
C
  104 FORMAT  (4F10.4)
  106 FORMAT  (3F10.4,2I5)
C
  108 FORMAT  (6F10.4)
  109 FORMAT  (6F10.3)
C
     RETURN
     END
```

```
C
      SUBROUTINE  PLOTZR(X , Y , N)
C
C
C        ***************************************************************
C                                                                      *
C        SUBROUTINE PLOTZR WILL PRODUCE A PLOT OF THE NODAL            *
C        POINTS OF ANY TWO DIMENSIONAL GEOMETRY . THE NODES            *
C        ARE REPRESENTED BY '*' , AND THE APPROPRIATE NODAL            *
C        POINT NUMBERS ARE PRINTED BELOW EACH NODE.                    *
C        PLOTZR(X,Y,N) ; X IS HORIZONTAL COORDINATE                    *
C                        Y IS VERTICAL   COORDINATE                    *
C                        N IS NUMBER OF NODES. (LESS THAN 1000)        *
C                                                                      *
C        ***************************************************************
C
C
      DIMENSION  X(N),Y(N),IDUM(104),JDUM(104),NU(10)
      DATA  NU/'1','2','3','4','5','6','7','8','9','0'/
      DATA  ISTAR/'*'/ , IBLANK/' '/ , IDOT/'.'/
C
      DO  20  I = 1 , 104
   20 IDUM(I) = IDOT
C
C
      WRITE  (6,85)  IDUM
   85 FORMAT  (1H1,  /,45X,'GRAPHICAL PRESENTATION OF NODAL ',
     1'POINTS.',/,48X,'Z  HORIZONTAL        R VERTICAL',/,20X,
     2104A1)
C
C
      XMAX = -1.0E 15
      YMAX = -1.0E 15
      XMIN = - XMAX
      YMIN = - YMAX
C
C
      DO  1  I = 1 , N
      IF  (X(I).GT.XMAX)   XMAX = X(I)
      IF  (X(I).LT.XMIN)   XMIN = X(I)
      IF  (Y(I).GT.YMAX)   YMAX = Y(I)
      IF  (Y(I).LT.YMIN)   YMIN = Y(I)
    1 CONTINUE
C
C
      DIFX = XMAX - XMIN
```

```
      DIFY = YMAX - YMIN                                                00009070
      FACTX = 100.0 / DIFX                                              00009080
      FACTY = 70.0 / DIFY                                               00009090
C                                                                       00009100
      DO  2  I = 1 , N                                                  00009110
      X(I) = FACTX * (X(I) - XMIN)                                      00009120
      Y(I) = 1.0 + FACTY * (Y(I) - YMIN)                                00009130
    2 CONTINUE                                                          00009140
C                                                                       00009150
      WRITE  (6,87)  YMAX                                               00009160
C                                                                       00009170
C                                                                       00009180
      JDUM(1) = IDOT                                                    00009190
      IDUM(103) = IBLANK                                                00009200
      K = 71                                                            00009210
      DO  9  I = 1 , 71                                                 00009220
      JDUM(104) = IDOT                                                  00009230
      DO  6  J = 2 , 103                                                00009240
      JDUM(J) = IBLANK                                                  00009250
    6 IDUM(J) = IBLANK                                                  00009260
C                                                                       00009270
C                                                                       00009280
      DO  7  J = 1 , N                                                  00009290
      M = Y(J)                                                          00009300
      IF  (M.EQ.K)  GO  TO  8                                           00009310
      GO  TO  7                                                         00009320
    8 NUM = X(J) + 2                                                    00009330
      IDUM(NUM) = ISTAR                                                 00009340
C                                                                       00009350
      K1 = J/100                                                        00009360
      IF  (K1.NE.0)  GO  TO  15                                         00009370
      K1 = J/10                                                         00009380
      IF  (K1.NE.0)  GO  TO  10                                         00009390
      JDUM(NUM) = NU(J)                                                 00009400
      GO  TO  7                                                         00009410
C                                                                       00009420
   10 K2 = J - 10*K1                                                    00009430
      JDUM(NUM) = NU(K1)                                                00009440
      IF  (K2.EQ.0)  K2 = 10                                            00009450
      JDUM(NUM+1) = NU(K2)                                              00009460
      GO  TO  7                                                         00009470
C                                                                       00009480
   15 K3 = J - 100*K1                                                   00009490
      K2 = K3 / 10                                                      00009500
      K3 = K3 - 10*K2                                                   00009510
      JDUM(NUM) = NU(K1)                                                00009520
      IF  (K2.EQ.0)  K2 = 10                                            00009530
      JDUM(NUM+1) = NU(K2)                                              00009540
```

```
      IF  (K3.EQ.0)  K3 = 10
      JDUM(NUM+2) = NU(K3)
C
    7 CONTINUE
C
C
      WRITE  (6,88)  IDUM
      WRITE  (6,88)  JDUM
      K = K - 1
    9 CONTINUE
C
C
      WRITE  (6,89)  YMIN
C
C
      DIF = (XMAX-XMIN) / 4.0
      X2 = XMIN + DIF
      X3 = X2 + DIF
      X4 = X3 + DIF
      WRITE  (6,90)  XMIN , X2 , X3 , X4 , XMAX
C
   87 FORMAT  (3X,'MAX.R=',F10.5)
   89 FORMAT  (3X,'MIN.R=',F10.5)
   88 FORMAT  (20X,104A1)
   90 FORMAT  (20X,'.+',24('.'),'+',24('.'),'+',24('.'),'+',
     124('.'),'+..',/,16X,4(F10.5,15X),F10.5,'   Z')
C
C
      WRITE  (6,555)
  555 FORMAT  (///,40X,'NOTE :',/,45X,'IF THE NODES ARE ',
     1'TOO CLOSED TO EACH OTHER',/,45X,'A NODE MAY NOT BE ',
     2'PRINTED , A NODE NUMBER',/,45X,'MAY BE MISPRINTED ,',
     3'OR BOTH.')
      RETURN
      END
```

```
C

      SUBROUTINE  PLOT(X,Y,LL)
C
      DIMENSION  X(75),Y(75),IDUM(102)
      DATA ISTAR/'*'/ , IPLUS/'+'/ , IBLANK/' '/ , IDOT/'.'/
C
C
C         *************************************************************
C                                                                     *
C         SUBROUTINE PLOT(X,Y,LL) WILL PLOT THE VARIATION OF          *
C         THE ENTRY TEMPERATURE VERSUS TIME AS 75 POINTS FOR          *
C         THE FIRST 150 TIME INCREMENTS,IF(TEMPERATURE.GE.0.)         *
C                X  IS TIME.                                          *
C                Y  IS TEMPERATURE.                                   *
C                LL=0  OUTSIDE FLOW.                                  *
C                LL=1  INSIDE  FLOW.                                  *
C                                                                     *
C         *************************************************************
C
C
      DO  1  I = 1 , 75
      IF  (Y(I).LT.0.D0)  GO  TO  50
    1 CONTINUE
      DO  5  I = 1 , 102
    5 IDUM(I) = IDOT
C
C
      YMAX = -1.0E 18
      YMIN = - YMAX
      DO  2  I = 1 , 75
      IF  (Y(I). GT .YMAX)  YMAX = Y(I)
      IF  (Y(I). LT .YMIN)  YMIN = Y(I)
    2 CONTINUE
C
C
      RAT = 1.0 + (100.0 * YMIN) / YMAX
      NUM = RAT
      IDUM(NUM) = IPLUS
C
C
      IF  (LL.EQ.0)  WRITE  (6,66)  IDUM
   66 FORMAT (1H1, / ,42X,'OUTSIDE FLUID ENTRY TIME TEMPERA'
     1'TURE PLOT',/,42X,41('*'), / ,7X,'TIME',5X,'TEMPERATURE',
     290X,'TEMPERATURE',/,29X,102A1)
C
C
```

```
      IF  (LL.NE.0)  WRITE  (6,77)  IDUM
   77 FORMAT (1H1, / ,42X,' INSIDE FLUID ENTRY TIME TEMPERA'
     1'TURE PLOT',/,43X,40('*'), / ,7X,'TIME',5X,'TEMPERATURE',
     290X,'TEMPERATURE',/,29X,102A1)
C
C
      DO  3  I = 1 , 101
    3 IDUM(I) = IBLANK
C
C
      DO  4  I = 1 , 75
      NUM = 1 + 100 * Y(I)/YMAX
      IDUM(NUM) = ISTAR
      WRITE  (6,88)  X(I),Y(I),IDUM
      IDUM(NUM) = IBLANK
    4 CONTINUE
   88 FORMAT  (5X,F8.2,4X,F10.3,'  .',102A1)
C
C
      DO  6  I = 1 , 101
    6 IDUM(I) = IDOT
      WRITE  (6,99) IDUM
   99 FORMAT  (29X,102A1)
   50 RETURN
      END
C
```

```
      SUBROUTINE  LLT  (N,A,WL)
C
C
C
C     **************************************************************
C                                                                  *
C     THIS SUBROUTINE WILL DECOMPOSE ANY  N BY N  REAL             *
C     SYMMETRIC MATRIX A INTO A LOWER TRIANGULAR MATRIX            *
C     'WL' OF THE SAME ORDER N , SUCH THAT THE PRODUCT             *
C     OF 'WL' AND ITS TRANSPOSE IS THE GIVEN MATRIX  A.            *
C     (THIS IS CHOLESKY DECOMPOSITION.)                            *
C                                                                  *
C     **************************************************************
C
C
      IMPLICIT  REAL *8 (A-H,O-Z)
      DIMENSION  A(N,N),WL(N,N)
      DO  7  I=1,N
      DO  8  J=1,N
    8 WL(I,J)=0.0D0
    7 CONTINUE
      WL(1,1)=DSQRT(A(1,1))
      DO  1  I=2,N
    1 WL(I,1)=A(I,1)/WL(1,1)
      N1=N-1
      M=3
      DO  2  I=2,N
      I1=I-1
      DO  3  J=1,I1
    3 WL(I,I)=WL(I,I)+WL(I,J)**2
      WL(I,I)=A(I,I)-WL(I,I)
      WL(I,I)=DSQRT(WL(I,I))
      IF (M.GT.N) GO TO  9
      DO  5  II=M,N
      DO  6  L=1,I1
    6 WL(II,I)=WL(II,I)+WL(I,L)*WL(II,L)
    5 WL(II,I)=(A(II,I)-WL(II,I))/WL(I,I)
    2 M=M+1
    9 RETURN
      END
```

```
C
      SUBROUTINE  CANDY
C
C
C        ***************************************************************
C                                                                      *
C        SUBROUTINE CANDY WILL CALCULATE CAPACITANCE MATRIX            *
C        C AND ADMITTANCE MATRIX Y IN THE BANDED FORM  THEN            *
C        THE ADDITIONAL YSTAR MATRIX IS EVALUATED AND ADDED            *
C        TO THE Y MATRIX .  A AND G  MATRICES ARE EVALUATED            *
C        AND PLACED IN THE  C AND Y  RESPECTIVELY.                     *
C        FINALLY,A CHOLESKY DECOMPOSITION IS PERFORMED ON C.           *
C                                                                      *
C        ***************************************************************
C
C
      IMPLICIT  REAL * 8 (A-H,O-Z)
C
      COMMON /ONE/ AJ(2,2),B(4,16),COPY(37,298),D(4,4),DL(4,4),
     1DNZR(2,8),ESM(16,16),F(24,298),SSM(298,66),STOR(20,150)
C
      COMMON /TWO/ AL(5),DENS(5),DETA(8),DXI(8),E(5),EPSO(4),
     1FE(16),GC(4),GX(4),POI(5),R(149),SHP(8),SHT(5),TK(5),Z(149)
C
      COMMON /TRE/ AXIALF,DTI,EVERY,FORS1,OMEGA,HTCI,HTCO,
     1POSI,PRES,Q1,Q2,Q3,SHMAX,SMAX,TIME,TIME1,TINIT,TMAXM,TREF,
     2TMAXO,RMAXM,RMAXO,ZMAXM,ZMAXO,Q4,BIG
C
      COMMON /FOR/ NN(40,9)
C
      COMMON /FIV/ NCFI(37),NCFO(37),NNR(37),NRE(37),NNL(37),
     1NPN(37)
      COMMON /SIX/ IBAND,IEXT,INTP,IVEC,JVEC,KK,LOAD,NCFIT,NCFOT,
     1NCHECK,NET,NDF,NGP,NNE,NNLT,NNRT,NPASS,NPROB,NRPPI,NRPPO,
     3NNT,NNRE,NPNT,IPLANE,NBAND
C
      EQUIVALENCE  (C,SSM) , (Y,SSM(1,18)) , (T,SSM(1,35)) ,
     1(BTE,SSM(1,36)) , (TEMP,SSM(1,37)) , (V,SSM(1,38)) ,
     2(FT1,SSM(1,39)) , (FT2,SSM(1,40)) , (CE,SSM(1,41)) ,
     3(YE,SSM(1,42)) , (OFF1,SSM(1,43)) , (OFF2,SSM(1,44)) ,
     4(DIAG,SSM(1,45)) , (BDRYI,SSM(1,46)) , (BDRYO,SSM(1,47)) ,
     5(ENTII,SSM(1,48)) , (ENTIO,SSM(1,49)), (ENTEI,SSM(1,50)),
     6(ENTEO,SSM(1,51)),(VOL,SSM(1,52)),(VOLO,SSM(1,53))
C
      DIMENSION  C(149,33),Y(149,33),T(149),BTE(149),TEMP(149),
     1V(149),FT1(90),FT2(90),CE(8,8),YE(8,8),OFF1(37),OFF2(37),
```

```
     2DIAG(37),BDRYI(16,5),BDRYO(16,5),VOL(36),VOLO(36),ENTEO(75),
     3ENTEI(75),ENTIO(75),ENTII(75)
C
      DO  12  I = 1 , NNT
      V(I) = 0.D0
      DO  12  J = 1 , NBAND
      C(I,J) = 0.D0
      Y(I,J) = 0.D0
   12 CONTINUE
C
C
      PAI = 3.141592653589793
      DO  25  L = 1 , NET
C
C
      M1 = NN(L,9)
      FACTC = 2.D0 * PAI * DENS(M1) * SHT(M1)
      FACTY = 2.D0 * PAI * TK(M1)
C
C
C
      DO  7  I = 1 , NNE
      DO  7  J = 1 , NNE
      CE(I,J) = 0.D0
    7 YE(I,J) = 0.D0
C
C
C
      DO  9  N = 1 , NGP
C
      ETA = GX(N)
C
      DO  9  M = 1 , NGP
C
      XI = GX(M)
C
C         SHAPE FUNCTIONS.
C
C
C
      SHP(1) = .25D0 * (1.D0-XI) * (1.D0-ETA) * (-XI-ETA-1.D0)
      SHP(2) = .5D0 * (1.D0-XI**2) * (1.D0-ETA)
      SHP(3) = .25D0 * (1.D0+XI) * (1.D0-ETA) * (XI-ETA-1.D0)
      SHP(4) = .5D0 * (1.D0-ETA**2) * (1.D0+XI)
      SHP(5) = .25D0 * (1.D0+XI) * (1.D0+ETA) * (XI+ETA-1.D0)
      SHP(6) = .5D0 * (1.D0-XI**2) * (1.D0+ETA)
      SHP(7) = .25D0 * (1.D0-XI) * (1.D0+ETA) * (-XI+ETA-1.D0)
      SHP(8) = .5D0 * (1.D0-ETA**2) * (1.D0-XI)
```

```
C
C
C
C      DERIVATIVES OF THE SHAPE FUNCTIONS WITH RESPECT TO XI.
C
C
      DXI(1)     = .25DO * (1.DO-ETA) * (2.DO*XI+ETA)
      DXI(2)     = -XI * (1.DO-ETA)
      DXI(3)     = .25DO * (1.DO-ETA) * (2.DO*XI-ETA)
      DXI(4)     = .5DO * (1.DO-ETA**2)
      DXI(5)     = .25DO * (1.DO+ETA) * (2.DO*XI+ETA)
      DXI(6)     = -XI * (1.DO+ETA)
      DXI(7)     = .25DO * (1.DO+ETA) * (2.DO*XI-ETA)
      DXI(8)     = -.5DO * (1.DO-ETA**2)
C
C
C      DERIVATIVES OF THE SHAPE FUNCTIONS WITH RESPECT TO ETA.
C
C
      DETA(1)    = .25DO * (1.DO-XI) * (2.DO*ETA+XI)
      DETA(2)    = -.5DO * (1.DO-XI**2)
      DETA(3)    = .25DO * (1.DO+XI) * (2.DO*ETA-XI)
      DETA(4)    = -ETA * (1.DO+XI)
      DETA(5)    = .25DO * (1.DO+XI) * (2.DO*ETA+XI)
      DETA(6)    = .5DO * (1.DO-XI**2)
      DETA(7)    = .25DO * (1.DO-XI) * (2.DO*ETA-XI)
      DETA(8)    = -ETA * (1.DO-XI)
C
C
C        CONSTRUCTION OF THE JACOBIAN MATRIX.
C
C
      DO   5   I = 1 , 2
      DO   5   J = 1 , 2
    5 AJ(I,J) = 0.DO
C
      DO   22  I = 1 , NNE
      AJ(1,1) = AJ(1,1) + DXI(I) * Z(NN(L,I))
      AJ(1,2) = AJ(1,2) + DXI(I) * R(NN(L,I))
      AJ(2,1) = AJ(2,1) + DETA(I) * Z(NN(L,I))
      AJ(2,2) = AJ(2,2) + DETA(I) * R(NN(L,I))
   22 CONTINUE
C
C
      DET = AJ(1,1) * AJ(2,2) - AJ(1,2) * AJ(2,1)
      DEI = 1.DO / DET
C
C        CONSTRUCTION OF INVERSE OF JACOBIAN
```

```
C
C
      DUM = AJ(1,1) * DEI
      AJ(1,1) =  AJ(2,2) * DEI
      AJ(1,2) = -AJ(1,2) * DEI
      AJ(2,1) = -AJ(2,1) * DEI
      AJ(2,2) =  DUM
C
      CR = 0.D0
      DO  21   I = 1 , NNE
      M1 = NN(L,I)
   21 CR = CR + SHP(I) * R(M1)
C
C
C         EVALUATION OF THE UPPER TRIANGLE OF  CE.
C
C
      DUM = FACTC * DET * CR * GC(N) * GC(M)
      DO  23   I = 1 , NNE
      DO  23   J = I , NNE
      CE(I,J) = CE(I,J) + DUM * SHP(I) * SHP(J)
   23 CONTINUE
C
C
C         CONSTRUCTION OF DNZR ;  THE MATRIX OF THE PARTIAL
C         DERIVATIVES OF SHAPE FUNCTIONS RESPECT TO Z AND R
C
C
      DO  30   I = 1 , NNE
      DNZR(1,I) = AJ(1,1) * DXI(I) + AJ(1,2) * DETA(I)
      DNZR(2,I) = AJ(2,1) * DXI(I) + AJ(2,2) * DETA(I)
   30 CONTINUE
C
C
C         EVALUATION OF UPPER TRIANGLE OF YE.
C
C
      DUM = FACTY * CR * DET * GC(N) * GC(M)
      DO  26   I = 1 , NNE
      DO  26   J = I , NNE
      DUM1 = DNZR(1,I) *DNZR(1,J)
      DUM2 = DNZR(2,I) * DNZR(2,J)
      YE(I,J) = YE(I,J) + (DUM1+DUM2) * DUM
   26 CONTINUE
C
C
    9 CONTINUE
C
```

```
C
      DO  27  I = 1 , NNE
      DO  27  J = 1 , NNE
      CE(J,I) = CE(I,J)
   27 YE(J,I) = YE(I,J)
C
C       GENERATING THE THERMAL CAPACITANCE AND ADMITTANCE
C       MATRICES IN THE BANDED FORM.
C
C
C
      DO  14  I = 1 , NNE
      II = NN(L,I) - 1
      DO  14  J = 1 , NNE
      J1 = NN(L,J) - 1
      IF  (II.GT.J1)  GO  TO  14
      J1 = J1 - II
      JJ = II+1
      KK = J1 + 1
      C(JJ,KK) = C(JJ,KK) + CE(I,J)
      Y(JJ,KK) = Y(JJ,KK) + YE(I,J)
   14 CONTINUE
C
C
   25 CONTINUE
C
C
      IF  (Q2.GT.0.D0)  GO  TO  40
      NC = 0
      GO  TO  50
   40 NC = NCFOT
      DO  64  I = 1 , NCFOT
      DIAG(I) = 0.D0
      OFF1(I) = 0.D0
   64 OFF2(I) = 0.D0
C
C
      NECFO = (NCFOT-1) / 2
      DO  60  M = 1 , NECFO
      K = 2 * M -1
C
      JL = NCFO(K  )
      JM = NCFO(K+1)
      JR = NCFO(K+2)
C
      DO  60  N = 1 , NGP
C
      XI = GX(N)
```

```
C
      SHP1    = -.5D0 * (1.D0-XI) * XI
      SHP2    =   1.D0 - XI**2
      SHP3    =  .5D0 * (1.D0+XI) * XI
C
      DXI1    = -.5D0 + XI
      DXI2    = -2.D0 * XI
      DXI3    =  .5D0 + XI
C
      RR = SHP3*R(JL) + SHP2*R(JM) + SHP1*R(JR)
C
      DRXI = DXI3*R(JL) + DXI2*R(JM) + DXI1*R(JR)
      DZXI = DXI3*Z(JL) + DXI2*Z(JM) + DXI1*Z(JR)
C
      DRXI = DRXI**2
      DZXI = DZXI**2
C
      DUMY = RR * GC(N) * DSQRT(DRXI+DZXI)
C
      DIAG( K ) = DIAG( K ) + SHP3*SHP3 * DUMY
      DIAG(K+1) = DIAG(K+1) + SHP2*SHP2 * DUMY
      DIAG(K+2) = DIAG(K+2) + SHP1*SHP1 * DUMY
      OFF1( K ) = OFF1( K ) + SHP3*SHP2 * DUMY
      OFF1(K+1) = OFF1(K+1) + SHP2*SHP1 * DUMY
      OFF2( K ) = OFF2( K ) + SHP1*SHP3 * DUMY
C
C
   60 CONTINUE
C
C
      CONST = 2.D0 * PAI * HTCO
C
      DO  67  I = 1 , NCFOT
      DIAG(I) = DIAG(I) * CONST
      OFF1(I) = OFF1(I) * CONST
   67 OFF2(I) = OFF2(I) * CONST
C
C
C         ADDING THE CONTRIBUTION OF Y STAR TO Y MATRIX
C         FOR  OUTSIDE CONVECTION
C
C
      DO  68  I = 1 , NCFOT
      K = NCFO(I)
   68 Y(K,1) = Y(K,1) + DIAG(I)
C
      N1 = NCFOT - 1
      DO  69  I = 1 , N1
```

```
      K1 = NCFO(I)
      K2 = NCFO(I+1)
      K = K2 - K1 + 1
      IF  (K.LE.0)   K = - K + 2
      K1 = MINO(K1,K2)
      Y(K1,K) = Y(K1,K) + OFF1(I)
   69 CONTINUE
C
      N2 = NCFOT-2
      DO  66  I = 1 , N2 , 2
      K1 = NCFO(I)
      K2 = NCFO(I+2)
      K = K2 - K1 + 1
      IF  (K.LE.0)   K = - K + 2
      K1 = MINO(K1,K2)
      Y(K1,K) = Y(K1,K) + OFF2(I)
   66 CONTINUE
C
C
C
   50 IF  (Q3.GT.0.D0)  GO  TO  41
      GO  TO  83
   41 DO  54  I = 1 , NCFIT
      J = NC + I
      DIAG(J) = 0.D0
      OFF1(J) = 0.D0
   54 OFF2(J) = 0.D0
C
C
      NECFI = (NCFIT - 1) / 2
      DO  59  M = 1 , NECFI
C
      K = 2 * M -1
C
      JL = NCFI(K)
      JM = NCFI(K+1)
      JR = NCFI(K+2)
C
      DO  59  N = 1 , NGP
C
      XI = GX(N)
C
C
      SHP1   = -.5D0 * (1.D0 - XI) * XI
      SHP2   =  1.D0 - XI**2
      SHP3   =  .5D0 * (1.D0 + XI) * XI
      DXI1   = -.5D0 + XI
      DXI2   = -2.D0 * XI
```

```
      DXI3    =   .5D0 + XI
C
      RR = SHP1*R(JL) + SHP2*R(JM) + SHP3*R(JR)
C
      DRXI  = DXI1*R(JL) + DXI2*R(JM) + DXI3*R(JR)
      DZXI  = DXI1*Z(JL) + DXI2*Z(JM) + DXI3*Z(JR)
C
      DRXI  = DRXI**2
      DZXI  = DZXI**2
C
      DUMY = RR * GC(N) * DSQRT(DRXI+DZXI)
C
      I = NC + K
C
      DIAG( I ) = DIAG( I )+ SHP1*SHP1 * DUMY
      DIAG(I+1) = DIAG(I+1)+ SHP2*SHP2 * DUMY
      DIAG(I+2) = DIAG(I+2)+ SHP3*SHP3 * DUMY
      OFF1( I ) = OFF1( I ) + SHP1*SHP2 * DUMY
      OFF1(I+1) = OFF1(I+1) + SHP2*SHP3 * DUMY
      OFF2( I ) = OFF2( I ) + SHP1*SHP3 * DUMY
C
C
   59 CONTINUE
C
C
      CONST = 2.D0 * PAI * HTCI
C
      DO   57  J = 1 , NCFIT
      I = NC + J
      DIAG(I) = DIAG(I) * CONST
      OFF1(I) = OFF1(I) * CONST
   57 OFF2(I) = OFF2(I) * CONST
C
C
C        ADDING THE CONTRIBUTION OF Y STAR TO Y MATRIX
C        FOR INSIDE CONVECTION
C
C
      DO   58  I = 1 , NCFIT
      J = NC + I
      K = NCFI(I)
   58 Y(K,1) = Y(K,1) + DIAG(J)
      M1 = NCFIT - 1
C
      DO   55  I = 1 , M1
      J = NC + I
      K1 = NCFI(I)
      K2 = NCFI(I+1)
```

```
      K = K2 - K1 + 1
      IF  (K.LE.0)   K = - K + 2
      K1 = MINO(K1,K2)
      Y(K1,K) = Y(K1,K) + OFF1(J)
   55 CONTINUE
C
      M2 = NCFIT - 2
      DO  56  I = 1 , M2 , 2
      J = NC + I
      K1 = NCFI(I)
      K2 = NCFI(I+2)
      K = K2 - K1 + 1
      IF  (K.LE.0)   K = - K + 2
      K1 = MINO(K1,K2)
      Y(K1,K) = Y(K1,K) + OFF2(J)
   56 CONTINUE
C
   83 CONTINUE
C
C
C
C        EVALUATION OF A AND G MATRICES
C
C
      DUM = .50D0 * DTI
      DO  1  I=1,NNT
      DO  1  J = 1 , NBAND
      EL = DUM * Y(I,J)
      C1 = C(I,J) + EL
      C2 = C(I,J) - EL
      C(I,J) = C1
      Y(I,J) = C2
    1 CONTINUE
C
C
C     C  NOW CONTAINS  C+.5*DTI*Y  AND  Y CONTAINS  C-.5*DTI*Y
C
C
C
C
C        THIS PART WILL DO THE CHOLESKY DECOMPOSITION OF
C        THE MATRIX <C> WHICH IS ALREADY PUT IN THE BANDED
C        FORM. ORIGINAL VERSION WAS CODED BY PROF. CANTIN
C
C
      DO  300  I = 1 , NNT
      DUM = C(I,1)
      DUM = DSQRT(DUM)
```

```
      DO  200  J = 1 , NBAND
  200 C(I,J) = C(I,J) / DUM
C
      DO  260  J = 2 , NBAND
      L = I + J - 1
      IF  (L.GT.NNT)  GO  TO  260
      AA = C(I,J)
      IF  (AA.EQ.0.D0)  GO  TO  260
      DO  250  K = J , NBAND
      M = K - J + 1
  250 C(L,M) = C(L,M) - AA * C(I,K)
  260 CONTINUE
  300 CONTINUE
C
C
      RETURN
      END
```

```
C
      SUBROUTINE  FLOW
C
C
C        ************************************************************
C                                                                    *
C        FOR A GIVEN OUTSIDE OR INSIDE ENTRY TIME VARIATION OF       *
C        FLUID TEMPERATURE , THE SUBROUTINE FLOW WILL EVALUATE       *
C        THE TEMPERATURE OF THE FLUID NODES AT A GIVEN TIME .        *
C        THE CALCULATION IS BASED UPON CONSTANT FLUX.                *
C                                                                    *
C        ************************************************************
C
C
      IMPLICIT  REAL * 8 (A-H,O-Z)
      REAL  * 4  ENTEO,ENTEI,ENTIO,ENTII
C
C
      COMMON /ONE/ AJ(2,2),B(4,16),COPY(37,298),D(4,4),DL(4,4),
     1DNZR(2,8),ESM(16,16),F(24,298),SSM(298,66),STOR(20,150)
C
      COMMON /TWO/ AL(5),DENS(5),DETA(8),DXI(8),E(5),EPSO(4),
     1FE(16),GC(4),GX(4),POI(5),R(149),SHP(8),SHT(5),TK(5),Z(149)
C
      COMMON /TRE/ AXIALF,DTI,EVERY,FORS1,OMEGA,HTCI,HTCO,
     1POSI,PRES,Q1,Q2,Q3,SHMAX,SMAX,TIME,TIME1,TINIT,TMAXM,TREF,
     2TMAXO,RMAXM,RMAXO,ZMAXM,ZMAXO,Q4,BIG
C
      COMMON /FOR/ NN(40,9)
C
      COMMON /FIV/ NCFI(37),NCFO(37),NNR(37),NRE(37),NNL(37),
     1NPN(37)
      COMMON /SIX/ IBAND,IEXT,INTP,IVEC,JVEC,KK,LOAD,NCFIT,NCFOT,
     1NCHECK,NET,NDF,NGP,NNE,NNLT,NNRT,NPASS,NPROB,NRPPI,NRPPO,
     3NNT,NNRE,NPNT,IPLANE,NBAND
C
      EQUIVALENCE  (C,SSM) , (Y,SSM(1,18)) , (T,SSM(1,35)) ,
     1(BTE,SSM(1,36)) , (TEMP,SSM(1,37)) , (V,SSM(1,38)) ,
     2(FT1,SSM(1,39)) , (FT2,SSM(1,40)) , (CE,SSM(1,41)) ,
     3(YE,SSM(1,42)) , (OFF1,SSM(1,43)) , (OFF2,SSM(1,44)) ,
     4(DIAG,SSM(1,45)) , (BDRYI,SSM(1,46)) , (BDRYO,SSM(1,47)) ,
     5(ENTII,SSM(1,48)) , (ENTIO,SSM(1,49)), (ENTEI,SSM(1,50)),
     6(ENTEO,SSM(1,51)),(VOL,SSM(1,52)),(VOLO,SSM(1,53))
C
      DIMENSION  C(149,33),Y(149,33),T(149),BTE(149),TEMP(149),
     1V(149),FT1(90),FT2(90),CE(8,8),YE(8,8),OFF1(37),OFF2(37),
```

```
     2DIAG(37),BDRYI(16,5),BDRYO(16,5),VOL(36),VOLO(36),ENTEO(75),
     3ENTEI(75),ENTIO(75),ENTII(75)
C
C
      C1 = .8115781021773632D0
      C2 = 1.675516081914557D0
      C3 = .02617993877991495D0
      C4 = 1.204277183876088D0
      C5 = -.2094395102393196
      C6 = -.3665191429188094
C
      PAI = 3.141592653589793
C
C        FLOW IN THE LONGITUDINAL DIRECTION , AS IN PIPE
C
C
      N = 0
      IF  (Q2.EQ.0.D0)  GO  TO  46
      N = NCFOT
      IF  (KK.EQ. 1)    GO  TO  40
      IF  (NCHECK.EQ.0) GO  TO  45
C
C
      DO  41  I = 1 , NCFOT
   41 FT1(I) = FT2(I)
C
   40 TIME = KK * DTI
C
C
      M = KK / 2
      I = KK - 2 * M
      M = M + 1
      IF ((KK.LE.150).AND.(I.EQ.1))  ENTEO(M) = FT1(1)
      IF ((KK.LE.150).AND.(I.EQ.1))  ENTIO(M) = TIME-DTI
C
C
      DO  42  I = 2 , NRPPO
   42 IF  (TIME.GT.BDRYO(I,1))  VEL = BDRYO(I,5)
C
      IF  (VEL.EQ.0.D0)  GO  TO  46
C
C
      IF  (KK.GT.1)  GO  TO  75
      NECFO = (NCFOT-1) / 2
C
C
      DUM = 0.D0
      DO  10  I=2,NCFOT
```

```
      R1 = R(NCFO(I-1))
      R2 = R(NCFO(I))
      IF  (R1.GE.R2)  DUM = R1
   10 CONTINUE
      IF  (DUM.EQ.0.D0)  DUM = R(NCFO(NCFOT))
C
C
C
      DUM = DUM + 1.D0
      AREAO = PAI * DUM**2
      VOLO(1) = 0.D0
      DO  70  I = 1 , NECFO
      K = 2 * I - 1
      J1 = NCFO( K )
      J2 = NCFO(K+1)
      J3 = NCFO(K+2)
C
      DUM = C1*R(J1)**2+C2*R(J2)**2+C3*R(J3)**2
      DUM=DUM+C4*R(J1)*R(J2)+C5*R(J1)*R(J3)+C6*R(J2)*R(J3)
      VOLO(K+1) = (AREAO-DUM)*DABS(Z(J2)-Z(J1))
C
C
      DUM = C3*R(J1)**2+C2*R(J2)**2+C1*R(J3)**2
      DUM=DUM+C6*R(J1)*R(J2)+C5*R(J1)*R(J3)+C4*R(J2)*R(J3)
      VOLO(K+2) = (AREAO-DUM)*DABS(Z(J3)-Z(J2))
   70 CONTINUE
C
C
   75 DO  73  I = 2 , NRPPO
      IF  (TIME.GT.BDRYO(I,1))  KK2 = I
   73 CONTINUE
      TIM = TIME
      TF = BDRYO(KK2,4) * (TIM-BDRYO(KK2,1))
      FT2(1) = BDRYO(KK2,2) + TF
C
C
      DUMY = VEL*(AREAO-PAI*R(NCFO(1))**2)
C
C
      DO  74  I = 2 , NCFOT
      TIM1 = VOLO(I) / DUMY
      TIM = TIM - TIM1
      DO  76  J = 2 , NRPPO
      IF  (TIM.GE.BDRYO(J,1))  KK2 = J
   76 CONTINUE
C
C
      TF = BDRYO(KK2,4) * (TIM-BDRYO(KK2,1))
```

```
      FT2(I) = BDRYO(KK2,2) + TF
   74 CONTINUE
      GO  TO  46
   45 TIME = KK * DTI
C
   46  IF  (Q3.EQ.0.D0)    GO  TO  56
C
C
      IF  (KK.EQ.1)      GO  TO  50
      IF(NCHECK.EQ.0)  GO  TO  55
C
      DO  51  I = 1 , NCFIT
      J = N + I
   51 FT1(J) = FT2(J)
C
   50 TIME = KK * DTI
C
C
      M = KK / 2
      I = KK - 2 * M
      M = M + 1
      IF ((KK.LE.150).AND.(I.EQ.1))  ENTEI(M) = FT1(N+1)
      IF ((KK.LE.150).AND.(I.EQ.1))  ENTII(M) = TIME-DTI
C
      DO  52  I = 2 , NRPPI
   52 IF  (TIME.GT.BDRYI(I,1))   VEL = BDRYI(I,5)
C
      IF  (VEL.EQ.0.D0)   GO  TO  56
C
      IF  (KK.GT. 1)  GO  TO  82
      NECFI = (NCFIT-1)/2
      AREA = PAI * R(NCFI(1))**2
      VOL(1) = 0.D0
C
C
C
      DO  81  I = 1 , NECFI
      K = 2 * I - 1
      J1 = NCFI( K )
      J2 = NCFI(K+1)
      J3 = NCFI(K+2)
C
      DUM = C1*R(J1)**2+C2*R(J2)**2+C3*R(J3)**2
      DUM=DUM+C4*R(J1)*R(J2)+C5*R(J1)*R(J3)+C6*R(J2)*R(J3)
      VOL(K+1) = DUM * DABS(Z(J2)-Z(J1))
C
C
      DUM = C3*R(J1)**2+C2*R(J2)**2+C1*R(J3)**2
```

```
      DUM=DUM+C6*R(J1)*R(J2)+C5*R(J1)*R(J3)+C4*R(J2)*R(J3)
      VOL(K+2) = DUM * DABS(Z(J3) - Z(J2))
   81 CONTINUE
C
C
C
C
C
   82 DO  83  I = 2 , NRPPI
      IF  (TIME. GT .BDRYI(I,1))  KK2 = I
   83 CONTINUE
C
C
      TIM = TIME
      TF = BDRYI(KK2,4) * (TIM-BDRYI(KK2,1))
      FT2(N+1) = BDRYI(KK2,2) + TF
C
C
      DUMY = VEL * AREA
C
C
      DO  84  I = 2 , NCFIT
      TIM1 = VOL(I) / DUMY
      TIM = TIM - TIM1
      DO  85  J = 2 , NRPPI
      IF  (TIM.GE.BDRYI(J,1))   KK2 = J
   85 CONTINUE
      TF = BDRYI(KK2,4) * (TIM-BDRYI(KK2,1))
      FT2(N+I) = BDRYI(KK2,2) + TF
   84 CONTINUE
      GO  TO  56
C
C
   55 TIME = KK * DTI
   56 RETURN
      END
```

```
C
      SUBROUTINE  FORMV
C
C
C        ************************************************************
C                                                                   *
C        THE VECTOR V OF THE RIGHT HAND SIDE OF THE DISCRETIZED     *
C        FINITE ELEMENT FORMULATION FOR CONVECTION OR  CONSTANT     *
C        TEMPERATURE (OUTSIDE OR INSIDE) BOUNDARY CONDITION  IS     *
C        FORMED HERE.                                               *
C                                                                   *
C        ************************************************************
C
C
      IMPLICIT  REAL * 8 (A-H,O-Z)
C
      COMMON /ONE/ AJ(2,2),B(4,16),COPY(37,298),D(4,4),DL(4,4),
     1DNZR(2,8),ESM(16,16),F(24,298),SSM(298,66),STOR(20,150)
C
      COMMON /TWO/ AL(5),DENS(5),DETA(8),DXI(8),E(5),EPSO(4),
     1FE(16),GC(4),GX(4),POI(5),R(149),SHP(8),SHT(5),TK(5),Z(149)
C
      COMMON /TRE/ AXIALF,DTI,EVERY,FORS1,OMEGA,HTCI,HTCO,
     1POSI,PRES,Q1,Q2,Q3,SHMAX,SMAX,TIME,TIME1,TINIT,TMAXM,TREF,
     2TMAXO,RMAXM,RMAXO,ZMAXM,ZMAXO,Q4,BIG
C
      COMMON /FOR/ NN(40,9)
C
      COMMON /FIV/ NCFI(37),NCFO(37),NNR(37),NRE(37),NNL(37),
     1NPN(37)
      COMMON /SIX/ IBAND,IEXT,INTP,IVEC,JVEC,KK,LOAD,NCFIT,NCFOT,
     1NCHECK,NET,NDF,NGP,NNE,NNLT,NNRT,NPASS,NPROB,NRPPI,NRPPO,
     3NNT,NNRE,NPNT,IPLANE,NBAND
C
      EQUIVALENCE  (C,SSM) , (Y,SSM(1,18)) , (T,SSM(1,35)) ,
     1(BTE,SSM(1,36)) , (TEMP,SSM(1,37)) , (V,SSM(1,38)) ,
     2(FT1,SSM(1,39)) , (FT2,SSM(1,40)) , (CE,SSM(1,41)) ,
     3(YE,SSM(1,42)) , (OFF1,SSM(1,43)) , (OFF2,SSM(1,44)) ,
     4(DIAG,SSM(1,45)) , (BDRYI,SSM(1,46)) , (BDRYO,SSM(1,47)) ,
     5(ENTII,SSM(1,48)) , (ENTIO,SSM(1,49)), (ENTEI,SSM(1,50)),
     6(ENTEO,SSM(1,51)),(VOL,SSM(1,52)),(VOLO,SSM(1,53))
C
      DIMENSION  C(149,33),Y(149,33),T(149),BTE(149),TEMP(149),
     1V(149),FT1(90),FT2(90),CE(8,8),YE(8,8),OFF1(37),OFF2(37),
     2DIAG(37),BDRYI(16,5),BDRYO(16,5),VOL(36),VOLO(36),ENTEO(75),
     3ENTEI(75),ENTIO(75),ENTII(75)
```

```
C
C
      N = 0
      IF  (Q2.GT.0.D0)   GO  TO  5
      GO  TO  6
    5 N = NCFOT
      DO  1  I = 1 , NCFOT
      J = NCFO(I)
    1 V(J) = 0.D0
C
      DO  2  J = 1 , NCFOT
      I = NCFO(J)
    2 V(I) = V(I) + DIAG(J) * (FT1(J)+FT2(J))
C
      N1 = NCFOT - 1
      DO  3  I = 1 , N1
      I1 = I +1
      K1 = NCFO( I )
      K2 = NCFO(I+1)
      V(K1) = V(K1) + OFF1(I) * (FT1(I1)+FT2(I1))
    3 V(K2) = V(K2) + OFF1(I) * (FT1(I )+FT2(I ))
C
      N2 = NCFOT - 2
      DO  4  I =  1 , N2 , 2
      I2 = I + 2
      K2 = NCFO(I+2)
      J = NCFO(I)
      V(J ) = V(J ) + OFF2(I) * (FT1(I2)+FT2(I2))
    4 V(K2) = V(K2) + OFF2(I) * (FT1(I)+FT2(I))
C
C
    6 IF  (Q3.GT.0.D0)  GO  TO  11
      GO  TO  12
   11 DO  7  I = 1 , NCFIT
      J = NCFI(I)
    7 V(J) = 0.D0
C
      DO  8  J = 1 , NCFIT
      I = NCFI(J)
      K = N + J
    8 V(I) = V(I) + DIAG(K) * (FT1(K)+FT2(K))
C
      N1 = NCFIT - 1
      DO  9  I = 1 , N1
      K = N + I
      I1 = K + 1
      K1 = NCFI( I )
      K2 = NCFI(I+1)
```

```
      V(K1) = V(K1) + OFF1(K) * (FT1(I1)+FT2(I1))
    9 V(K2) = V(K2) + OFF1(K) * (FT1(K )+FT2(K ))
C
      N2 = NCFIT - 2
      DO  10  I = 1 , N2 , 2
      K = N + I
      I2 = K + 2
      J = NCFI( I )
      K2 = NCFI(I+2)
      V(J ) = V(J ) + OFF2(K) * (FT1(I2)+FT2(I2))
   10 V(K2) = V(K2) + OFF2(K) * (FT1(K)+FT2(K))
C
   12 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE  TEMPER
C
C
C        ****************************************************************
C                                                                       *
C        IN THIS SUBROUTINE THE NODAL TEMPERATURES  ARE                 *
C        EVALUATED BY TRAPEZOIDAL INTEGRATION. ALSO FOR                 *
C        EVERY 10 STEPS OF TIME INTEGRATION THE  'IRONS                 *
C        CORRECTION'  IS APPLIED .                                      *
C                                                                       *
C        ****************************************************************
C
C
      IMPLICIT  REAL * 8 (A-H,O-Z)                       .
C
      COMMON /ONE/ AJ(2,2),B(4,16),COPY(37,298),D(4,4),DL(4,4),
     1DNZR(2,8),ESM(16,16),F(24,298),SSM(298,66),STOR(20,150)
C
      COMMON /TWO/ AL(5),DENS(5),DETA(8),DXI(8),E(5),EPSO(4),
     1FE(16),GC(4),GX(4),POI(5),R(149),SHP(8),SHT(5),TK(5),Z(149)
C
      COMMON /TRE/ AXIALF,DTI,EVERY,FORS1,OMEGA,HTCI,HTCO,
     1POSI,PRES,Q1,Q2,Q3,SHMAX,SMAX,TIME,TIME1,TINIT,TMAXM,TREF,
     2TMAXO,RMAXM,RMAXO,ZMAXM,ZMAXO,Q4,BIG
C
      COMMON /FOR/ NN(40,9)
C
      COMMON /FIV/ NCFI(37),NCFO(37),NNR(37),NRE(37),NNL(37),
     1NPN(37)
      COMMON /SIX/ IBAND,IEXT,INTP,IVEC,JVEC,KK,LOAD,NCFIT,NCFOT,
     1NCHECK,NET,NDF,NGP,NNE,NNLT,NNRT,NPASS,NPROB,NRPPI,NRPPO,
     3NNT,NNRE,NPNT,IPLANE,NBAND
C
      EQUIVALENCE  (C,SSM) , (Y,SSM(1,18)) , (T,SSM(1,35)) ,
     1(BTE,SSM(1,36)) , (TEMP,SSM(1,37)) , (V,SSM(1,38)) ,
     2(FT1,SSM(1,39)) , (FT2,SSM(1,40)) , (CE,SSM(1,41)) ,
     3(YE,SSM(1,42)) , (OFF1,SSM(1,43)) , (OFF2,SSM(1,44)) ,
     4(DIAG,SSM(1,45)) , (BDRYI,SSM(1,46)) , (BDRYO,SSM(1,47)) ,
     5(ENTII,SSM(1,48)) , (ENTIO,SSM(1,49)), (ENTEI,SSM(1,50)),
     6(ENTEO,SSM(1,51)),(VOL,SSM(1,52)),(VOLO,SSM(1,53))
C
      DIMENSION  C(149,33),Y(149,33),T(149),BTE(149),TEMP(149),
     1V(149),FT1(90),FT2(90),CE(8,8),YE(8,8),OFF1(37),OFF2(37),
     2DIAG(37),BDRYI(16,5),BDRYO(16,5),VOL(36),VOLO(36),ENTEO(75),
     3ENTEI(75),ENTIO(75),ENTII(75)
C
```

```
C
C
      NCHECK = 1
C
      DUM = DTI / 2.D0
      DO  1  I=1,NNT
    1 V(I) = DUM * V(I)
C
C
C         THIS PART WILL MULTIPLY THE BANDED MATRIX Y BY THE
C         COLUMN VECTOR OF TEMPERATURE AND ADD TO V.
C
      DO  7  J = 1 , NNT
      BTE(J) = 0.D0
      M = MIN0(NBAND,NNT+1-J)
      DO  5  K = 1 , M
      L = J + K - 1
      BTE(J) = BTE(J) + Y(J,K) * TEMP(L)
    5 CONTINUE
C
      IF  (J. LT .2)  GO  TO  7
      L = MIN0(J , NBAND)
      DO  6  K = 2 , L
      M = J - K + 1
      BTE(J) = BTE(J) + Y(M,K) * TEMP(M)
    6 CONTINUE
    7 BTE(J) = BTE(J) + V(J)
C
C
C
C         SOLVING FOR TEMPERATURE BY FORWARD AND BACK SUBSTITUTION
C
C
C
      DO  32  I = 1 , NNT
      BTE(I) = BTE(I) / C(I,1)
      DO  31  J = 2 , NBAND
      L = I + J - 1
      IF  (L. GT .NNT)  GO  TO  32
      DUM = C(I,J)
   31 BTE(L) = BTE(L) - DUM * BTE(I)
   32 CONTINUE
C
      BTE(NNT) = BTE(NNT) / C(NNT,1)
      DO  34  L = 2 , NNT
      K = NNT - L + 1
      SUM = 0.D0
      DO  33  J = 2 , NBAND
```

```
      M = J + K - 1
      IF  (NNT. LT .M)    GO  TO  34
   33 SUM = SUM + C(K,J) * BTE(M)
   34 BTE(K) = (BTE(K) -SUM) / C(K,1)
C
C
      IF  (KK.GT.2)  GO  TO  2
      KO = 0
C
C        APPLICATION OF IRONS CORRECTION.
C
    2 K1 = KO + 12 - KK
      GO  TO (70, 60 ,50) ,K1
      GO  TO  14
C
   50 DO  51  I = 1 , NNT
   51 T(I) = .25DO * BTE(I)
      GO  TO  14
C
   60 DO  61  I = 1 , NNT
   61 T(I) = T(I) + .5DO * BTE(I)
      GO  TO  14
C
   70 DO  71  I = 1 , NNT
   71 TEMP(I) = T(I) + .25DO * BTE(I)
C
C     TEMPERATURE IS CORRECTED NOW
C
      NCHECK = 0
      KO = KO + 10
      KK = KK - 1
      TIME = KK*DTI
      IP = 0
C
      GO  TO  16
   14 DO  4  I = 1,NNT
    4 TEMP(I) = BTE(I)
      IP = 1
C
   16 CONTINUE
C
      C1 = 1000.DO * TIME
      DO  21  J = 1 , IVEC
      C2 =(TIME1 + EVERY * (J-1)) * 1000.DO
      DC=DABS(C1-C2)
      IF  (DC.LE..0001)  GO  TO  20
   21 CONTINUE
C
```

```
      GO  TO  22
C
C
   20 DO  40   I = 1 , NNT
   40 STOR(J,I) = TEMP(I)
      STOR(J,NNT+1) = TIME
C
C
   22 IF  (INTP.EQ.0)  GO  TO  81
      INT1 = INTP
   83 KUNT = KK / INT1
      IF  (KUNT.EQ.0)  GO  TO  81
      KUNT = KK - KUNT * INT1
C
      IF  (KUNT.EQ.0)  GO  TO  82
      INT1 = INT1 + INTP
      GO  TO  83
C
   82 ID = INTP / 10
      ID = INTP - ID * 10
      IF  ((ID.EQ.0).AND.(IP.NE.0))  GO TO 81
      WRITE  (6,101)  TIME
  101 FORMAT (/,5X,'AT TIME=',F8.2,' THE TEMPERATURES AT THE ',
     1'NODES ARE :',/)
      WRITE  (6,222)  (I,TEMP(I),I=1,NNT)
  222 FORMAT  (7(1X,'(',I3,')',1PD13.5))
   81 KK = KK + 1
C
C
      RETURN
      END
C
```

```
      SUBROUTINE  STIFF
C
C
C        ***************************************************************
C                                                                      *
C        SUBROUTINE STIFF , CALCULATES THE STIFFNESS MATRIX            *
C        FOR A GIVEN PROBLEM IN THE BANDED FORM,APPLIES THE            *
C        REQUIRED BOUNDARY CONDITIONS AND AT THE END  MAKES            *
C        THE CHOLESKY DECOMPOSITION.                                   *
C                                                                      *
C        ***************************************************************
C
C
      IMPLICIT  REAL * 8 (A-H,O-Z)
C
      COMMON /ONE/ AJ(2,2),B(4,16),COPY(37,298),D(4,4),DL(4,4),
     1DNZR(2,8),ESM(16,16),F(24,298),SSM(298,66),STOR(20,150)
C
      COMMON /TWO/ AL(5),DENS(5),DETA(8),DXI(8),E(5),EPSO(4),
     1FE(16),GC(4),GX(4),POI(5),R(149),SHP(8),SHT(5),TK(5),Z(149)
C
      COMMON /TRE/ AXIALF,DTI,EVERY,FORS1,OMEGA,HTCI,HTCO,
     1POSI,PRES,Q1,Q2,Q3,SHMAX,SMAX,TIME,TIME1,TINIT,TMAXM,TREF,
     2TMAXO,RMAXM,RMAXO,ZMAXM,ZMAXO,Q4,BIG
C
      COMMON /FOR/ NN(40,9)
C
      COMMON /FIV/ NCFI(37),NCFO(37),NNR(37),NRE(37),NNL(37),
     1NPN(37)
      COMMON /SIX/ IBAND,IEXT,INTP,IVEC,JVEC,KK,LOAD,NCFIT,NCFOT,
     1NCHECK,NET,NDF,NGP,NNE,NNLT,NNRT,NPASS,NPROB,NRPPI,NRPPO,
     3NNT,NNRE,NPNT,IPLANE,NBAND
C
C
C
      DO  5  I = 1 , NDF
      DO  5  J = 1 , IBAND
    5 SSM(I,J) = 0.D0
      PAI = 3.141592653589793
      DO  31  L = 1 , NET
      IF  (L. EQ .1 )  GO  TO  1
      IF  (NN(L,9).EQ.NN(L-1,9))  GO  TO  3
C
    1 K = NN(L,9)
      EL = 1.D0 - POI(K)
      FACT = (E(K)*EL)/((1.D0+POI(K))*(1.D0-2.D0*POI(K)))
```

```
      EL = POI(K) / EL
C
      D(1,1) = FACT
      D(1,2) = EL * FACT
      D(1,3) = D(1,2)
      D(1,4) = 0.D0
      D(2,2) = FACT
      D(2,3) = D(1,2)
      D(2,4) = 0.D0
      D(3,3) = FACT
      D(3,4) = 0.D0
      D(4,4) = E(K)/(2.D0*(1.D0+POI(K)))
C
C
      DO  27  I = 1 , 4
      DO  27  J = I , 4
   27 D(J,I) = D(I,J)
C
C
      CALL  LLT  (4,D,DL)
C
C
C     DL  IS THE LOWER TRIANGLE OF DECOMPOSED D.
C
C
    3 CONTINUE
C
      NNE2 = 2 * NNE
      DO  11  I = 1 , NNE2
      DO  11  J = 1 , NNE2
   11 ESM(I,J) = 0.D0
C
C
      DO  9  N = 1,NGP
      ETA = GX(N)
      DO  8  M = 1,NGP
      XI = GX(M)
C
C
      DO  7  I = 1,4
      DO  7  J = 1,NNE2
    7 B(I,J) = 0.D0
C
C
C
      SHP(1) = .25D0 * (1.D0-XI) * (1.D0-ETA) * (-XI-ETA-1.D0)
      SHP(2) = .5D0 * (1.D0-XI**2) * (1.D0-ETA)
      SHP(3) = .25D0 * (1.D0+XI) * (1.D0-ETA) * (XI-ETA-1.D0)
```

```
      SHP(4) = .5D0 * (1.D0-ETA**2) * (1.D0+XI)
      SHP(5) = .25D0 * (1.D0+XI) * (1.D0+ETA) * (XI+ETA-1.D0)
      SHP(6) = .5D0 * (1.D0-XI**2) * (1.D0+ETA)
      SHP(7) = .25D0 * (1.D0-XI) * (1.D0+ETA) * (-XI+ETA-1.D0)
      SHP(8) = .5D0 * (1.D0-ETA**2) * (1.D0-XI)
C
C
C      DERIVATIVES OF THE SHAPE FUNCTIONS WITH RESPECT TO XI.
C
C
      DXI(1)      = .25D0 * (1.D0-ETA) * (2.D0*XI+ETA)
      DXI(2)      = -XI * (1.D0-ETA)
      DXI(3)      = .25D0 * (1.D0-ETA) * (2.D0*XI-ETA)
      DXI(4)      = .5D0 * (1.D0-ETA**2)
      DXI(5)      = .25D0 * (1.D0+ETA) * (2.D0*XI+ETA)
      DXI(6)      = -XI * (1.D0+ETA)
      DXI(7)      = .25D0 * (1.D0+ETA) * (2.D0*XI-ETA)
      DXI(8)      = -.5D0 * (1.D0-ETA**2)
C
C
C      DERIVATIVES OF THE SHAPE FUNCTIONS WITH RESPECT TO ETA.
C
C
      DETA(1)     = .25D0 * (1.D0-XI) * (2.D0*ETA+XI)
      DETA(2)     = -.5D0 * (1.D0-XI**2)
      DETA(3)     = .25D0 * (1.D0+XI) * (2.D0*ETA-XI)
      DETA(4)     = -ETA * (1.D0+XI)
      DETA(5)     = .25D0 * (1.D0+XI) * (2.D0*ETA+XI)
      DETA(6)     = .5D0 * (1.D0-XI**2)
      DETA(7)     = .25D0 * (1.D0-XI) * (2.D0*ETA-XI)
      DETA(8)     = -ETA * (1.D0-XI)
C
C
C         CONSTRUCTION OF THE JACOBIAN MATRIX.
C
C
      DO   20   I = 1 , 2
      DO   20   J = 1 , 2
   20 AJ(I,J) = 0.D0
C
C
      DO   22   I = 1 , NNE
      AJ(1,1) = AJ(1,1) + DXI(I)   * Z(NN(L,I))
      AJ(1,2) = AJ(1,2) + DXI(I)   * R(NN(L,I))
      AJ(2,1) = AJ(2,1) + DETA(I)  * Z(NN(L,I))
      AJ(2,2) = AJ(2,2) + DETA(I)  * R(NN(L,I))
   22 CONTINUE
C
```

```
C
      DET =AJ(1,1) * AJ(2,2) - AJ(1,2) * AJ(2,1)
      DEI =1.0D0/DET
C
C
C     INVERTING THE JACOBIAN AND STORING IN  AJ  AGAIN.
C
C
      DUM = AJ(1,1) * DEI
      AJ(1,1) = AJ(2,2) * DEI
      AJ(1,2) = -AJ(1,2) * DEI
      AJ(2,1) = -AJ(2,1) * DEI
      AJ(2,2) = DUM
C
C
C
      CR = 0.D0
      DO  2  I = 1,NNE
      M1 = NN(L,I)
    2 CR = CR + SHP(I) * R(M1)
C
C
C     EVALUATION OF THE B MATRIX
C
C
      DO  15  I = 1, NNE
      K1 = 2 * I - 1
      K2 = K1 + 1
      B(1,K2) = AJ(1,1) * DXI(I) + AJ(1,2) * DETA(I)
      B(2,K1) = AJ(2,1) * DXI(I) + AJ(2,2) * DETA(I)
      B(3,K1) = SHP(I) / CR
      B(4,K1) = B(1,K2)
      B(4,K2) = B(2,K1)
   15 CONTINUE
C
C
C     CALCULATION OF (DL TRANSPOSE)*B AND STORING IN <B>
C
C
      DO  16  I = 1 ,4
      DO  13  J = 1,NNE2
      DUM = 0.0D0
      DO  17  K = I,4
   17 DUM = DUM + DL(K,I) * B(K,J)
   13 B(I,J) = DUM
   16 CONTINUE
C
C
```

```
C     NOTE : <B> NOW CONTAINS THE PRODUCT OF (DL TRANSPOSE)*B
C
C     CALCULATION OF <BT>*<D>*<B> AND ELEMENT STIFFNESS MATRIX
C                                   <E S M >
C
C
      DUM1 = 2.D0 * PAI * CR * DET * GC(N) * GC(M)
      DO  19  I = 1,NNE2
      DO  19  J = 1, I
      G = 0.0D0
      DO  18  K = 1 ,4
   18 G = G + B(K,I) * B(K,J)
      ESM(I,J) = ESM(I,J) + G * DUM1
   19 CONTINUE
C
C
      DO  24  I = 1 , NNE2
      DO  24  J = 1 , I
   24 ESM(J,I) = ESM(I,J)
C
C
    8 CONTINUE
    9 CONTINUE
C
C
C
C             CONSTRUCTION OF  THE SYSTEM STIFFNESS MATRIX
C             IN BANDED FORM.
C
C
      DO  25  I = 1 , NNE
      II = 2 * (NN(L,I) -1)
C
      DO  25  J = 1 , NNE
      J1 = 2 * (NN(L,J) - 1)
      IF  (II.GT.J1)  GO  TO  25
      J1 = J1 - II
C
      DO  25  K = 1 , 2
      JJ = II + K
      L1 = 2 * (I-1) + K
      IJ = 1
      IF  (J1.EQ.0)  IJ = K
      DO  25  JI = IJ , 2
      M1 = 2 * (J-1) + JI
      KK = J1 + JI - K + 1
      SSM(JJ,KK) = SSM(JJ,KK) + ESM(L1,M1)
   25 CONTINUE
```

```
C
   31 CONTINUE
C
      IF (IPLANE.EQ.0)  GO  TO  49
C
C
C        PART OF SSM IS BEING STORED IN COPY.
C
C
      DO  41  J = 1 , NDF
      DO  41  I = 1 , NNRE
   41 COPY(I,J) = 0.D0
C
C
      DO  44  II = 1 , NNRE
      K = 2 * NRE(II)
      JJ = K
      DO  42  J = 1 , IBAND
      IF  (JJ.GT.NDF)  GO  TO  45
      COPY(II,JJ) = SSM(K,J)
   42 JJ = JJ + 1
C
C
   45 LL = K - 1
      IF  (LL.EQ.0)  GO  TO  44
      JJ = LL
C
C
      DO  43  J = 1 , LL
      IF  (J.EQ.IBAND)  GO  TO  44
      COPY(II,JJ) = SSM(JJ,J+1)
   43 JJ = JJ - 1
   44 CONTINUE
C
C
C
   49 CONTINUE
C
      IF  (NNRT.EQ.0)  GO  TO  33
      DO  34  I = 1 , NNRT
      K = 2 * NNR(I) - 1
   34 SSM(K,1) = SSM(K,1) * BIG
C
   33 IF  (NNLT.EQ.0)  GO  TO  35
      DO  36  I = 1 , NNLT
      K = 2 * NNL(I)
   36 SSM(K,1) = SSM(K,1) * BIG
   35 CONTINUE
```

```
C
C
C
C        DECOMPOSING SYSTEM STIFFNESS MATRIX
C
      DO   300  I = 1 , NDF
      DUM = SSM(I,1)
      DUM = DSQRT(DUM)
      DO   200  J = 1 , IBAND
  200 SSM(I,J) = SSM(I,J) / DUM
C
      DO   260  J = 2 , IBAND
      L = I + J -1
      IF   (L.GT.NDF)  GO  TO  260
      AA = SSM(I,J)
      IF   (AA.EQ.0.DO)  GO  TO  260
      DO   250  K = J , IBAND
      M = K - J + 1
  250 SSM(L,M) = SSM(L,M) - AA * SSM(I,K)
  260 CONTINUE
  300 CONTINUE
  350 RETURN
      END
```

```
      SUBROUTINE  FORMF
C
C
C        ************************************************************
C                                                                   *
C        THE THERMAL LOAD VECTORS ARE BEING CALCULATED HERE         *
C        AND FOR THE BOUNDARY CONDITION "ENDS REMAIN PLANE"         *
C        ANOTHER LOAD VECTOR FOR THE UNIT END  DISPLACEMENT         *
C        IS BEING CALCULATED.                                       *
C                                                                   *
C        ************************************************************
C
C
      IMPLICIT  REAL * 8 (A-H,O-Z)
C
      COMMON /ONE/ AJ(2,2),B(4,16),COPY(37,298),D(4,4),DL(4,4),
     1DNZR(2,8),ESM(16,16),F(24,298),SSM(298,66),STOR(20,150)
C
      COMMON /TWO/ AL(5),DENS(5),DETA(8),DXI(8),E(5),EPSO(4),
     1FE(16),GC(4),GX(4),POI(5),R(149),SHP(8),SHT(5),TK(5),Z(149)
C
      COMMON /TRE/ AXIALF,DTI,EVERY,FORS1,OMEGA,HTCI,HTCO,
     1POSI,PRES,Q1,Q2,Q3,SHMAX,SMAX,TIME,TIME1,TINIT,TMAXM,TREF,
     2TMAXO,RMAXM,RMAXO,ZMAXM,ZMAXO,Q4,BIG
C
      COMMON /FOR/ NN(40,9)
C
      COMMON /FIV/ NCFI(37),NCFO(37),NNR(37),NRE(37),NNL(37),
     1NPN(37)
      COMMON /SIX/ IBAND,IEXT,INTP,IVEC,JVEC,KK,LOAD,NCFIT,NCFOT,
     1NCHECK,NET,NDF,NGP,NNE,NNLT,NNRT,NPASS,NPROB,NRPPI,NRPPO,
     3NNT,NNRE,NPNT,IPLANE,NBAND
C
      PAI = 3.141592653589793
      IDUM = IVEC + JVEC
      IF  (LOAD.NE.0)  IDUM = IDUM -1
      DO  1  I = 1 , IDUM
      DO  1  J = 1 , NDF
    1 F(I,J) = 0.D0
C
      IDUM = IDUM + 1
      IF  (LOAD.NE.0)  IDUM = IDUM +1
      DO  4  I = 1 , NDF
    4 F(IDUM,I) = 0.D0
C
      NNE2 = 2 * NNE
```

```
C
      DO  25  L = 1 , NET
C
C
C
      DO  9  N = 1,NGP
      ETA = GX(N)
      DO  8  M = 1,NGP
      XI = GX(M)
C
C
      DO  7  I = 1,4
      DO  7  J = 1,NNE2
    7 B(I,J) = 0.DO
C
C
C
      SHP(1) = .25DO * (1.DO-XI) * (1.DO-ETA) * (-XI-ETA-1.DO)
      SHP(2) = .5DO * (1.DO-XI**2) * (1.DO-ETA)
      SHP(3) = .25DO * (1.DO+XI) * (1.DO-ETA) * (XI-ETA-1.DO)
      SHP(4) = .5DO * (1.DO-ETA**2) * (1.DO+XI)
      SHP(5) = .25DO * (1.DO+XI) * (1.DO+ETA) * (XI+ETA-1.DO)
      SHP(6) = .5DO * (1.DO-XI**2) * (1.DO+ETA)
      SHP(7) = .25DO * (1.DO-XI) * (1.DO+ETA) * (-XI+ETA-1.DO)
      SHP(8) = .5DO * (1.DO-ETA**2) * (1.DO-XI)
C
C
C      DERIVATIVES OF THE SHAPE FUNCTIONS WITH RESPECT TO XI.
C
C
      DXI(1)      = .25DO * (1.DO-ETA) * (2.DO*XI+ETA)
      DXI(2)      = -XI * (1.DO-ETA)
      DXI(3)      = .25DO * (1.DO-ETA) * (2.DO*XI-ETA)
      DXI(4)      = .5DO * (1.DO-ETA**2)
      DXI(5)      = .25DO * (1.DO+ETA) * (2.DO*XI+ETA)
      DXI(6)      = -XI * (1.DO+ETA)
      DXI(7)      = .25DO * (1.DO+ETA) * (2.DO*XI-ETA)
      DXI(8)      = -.5DO * (1.DO-ETA**2)
C
C
C      DERIVATIVES OF THE SHAPE FUNCTIONS WITH RESPECT TO ETA.
C
C
      DETA(1)     = .25DO * (1.DO-XI) * (2.DO*ETA+XI)
      DETA(2)     = -.5DO * (1.DO-XI**2)
      DETA(3)     = .25DO * (1.DO+XI) * (2.DO*ETA-XI)
      DETA(4)     = -ETA * (1.DO+XI)
      DETA(5)     = .25DO * (1.DO+XI) * (2.DO*ETA+XI)
```

```
      DETA(6)     = .5D0 * (1.D0-XI**2)
      DETA(7)     = .25D0 * (1.D0-XI) * (2.D0*ETA-XI)
      DETA(8)     = -ETA * (1.D0-XI)
C
C
C        CONSTRUCTION OF THE JACOBIAN MATRIX.
C
C
      DO  20  I = 1 , 2
      DO  20  J = 1 , 2
   20 AJ(I,J) = 0.D0
C
C
      DO  22  I = 1 , NNE
      AJ(1,1) = AJ(1,1) + DXI(I)  * Z(NN(L,I))
      AJ(1,2) = AJ(1,2) + DXI(I)  * R(NN(L,I))
      AJ(2,1) = AJ(2,1) + DETA(I) * Z(NN(L,I))
      AJ(2,2) = AJ(2,2) + DETA(I) * R(NN(L,I))
   22 CONTINUE
C
      DET =AJ(1,1) * AJ(2,2) - AJ(1,2) * AJ(2,1)
      DEI =1.0D0/DET
C
C
C     INVERTING THE JACOBIAN AND STORING IN  AJ  AGAIN.
C
C
      DUM = AJ(1,1) * DEI
      AJ(1,1) = AJ(2,2) * DEI
      AJ(1,2) = -AJ(1,2) * DEI
      AJ(2,1) = -AJ(2,1) * DEI
      AJ(2,2) = DUM
C
C
      CR = 0.D0
      DO  2  I = 1,NNE
      M1 = NN(L,I)
    2 CR = CR + SHP(I) * R(M1)
C
C
C
C     EVALUATION OF THE B MATRIX
C
C
      DO  15  I = 1, NNE
      K1 = 2 * I - 1
      K2 = K1 + 1
      B(1,K2) = AJ(1,1) * DXI(I) + AJ(1,2) * DETA(I)
```

```
      B(2,K1) = AJ(2,1) * DXI(I) + AJ(2,2) * DETA(I)
      B(3,K1) = SHP(I) / CR
      B(4,K1) = B(1,K2)
      B(4,K2) = B(2,K1)
   15 CONTINUE
C
C
      DUM1 = 2.D0 * PAI * CR * DET * GC(N) * GC(M)
C
C
      DO  30  KK = 1 , IVEC
C
C
C        EVALUATING TEMPERATURE AT GAUSS POINT
C
      TGP = 0.D0
      DO  3  I = 1 , NNE
      M1 = NN(L , I)
    3 TGP = TGP + SHP(I) * (STOR(KK,M1)-TREF)
C
      TGP = AL(NN(L,9)) * TGP
C
C
C        CALCULATION OF ELEMENT INITIAL STRAIN VECTOR <EPSO>
C
C
      DO  6  I = 1 , 3
    6 EPSO(I) = TGP
C
C
C        NOTE  EPSO(4) =0.D0
C
C        PRODUCT OF <BTRANSPOSE> * <D> * <EPSO>
C        CALCULATION OF ELEMENT LOAD VECTOR ,
C
      DUM = (TGP*E(NN(L,9)))/(1.D0-2.D0*POI(NN(L,9)))
C
      DUM = DUM * DUM1
      DO  17  I = 1 , NNE2 , 2
      J = I + 1
      FE(I) = (B(2,I) + B(3,I)) * DUM
   17 FE(J) = B(1,J) * DUM
C
C
C        CONSTRUCTION OF SYSTEM LOAD VECTORS   F
C
C
      DO  18  I = 1 , NNE
```

```
      I2 = 2 * I - 1
      I1 = NN(L,I) * 2 - 1
      F(KK,I1) = F(KK,I1) + FE(I2)
      F(KK,I1+1) = F(KK,I1+1) + FE(I2+1)
C
C
   18 CONTINUE
   30 CONTINUE
    8 CONTINUE
    9 CONTINUE
   25 CONTINUE
C
C
      IF (IPLANE.EQ.0)  GO  TO  40
C
C        LOAD VECTOR FOR UNIT END DISPLACEMENT
C
      K = IDUM
C
      DO  41  I = 1 , NNRE
      L = 2 * NRE(I)
   41 F(K,L) = COPY(I,L) * BIG
C
   40 CONTINUE
      RETURN
      END
C
C
```

```
      SUBROUTINE  CENTF
C
C
C       ************************************************************
C                                                                  *
C       SUBROUTINE CENTF  CALCULATES THE CENTRIFUGAL LOAD          *
C       VECTOR FOR A GIVEN NUMBER OF REVOLUTIONS PER MINUTE        *
C                                                                  *
C       ************************************************************
C
C
      IMPLICIT  REAL * 8 (A-H,O-Z)
C
C
      COMMON /ONE/ AJ(2,2),B(4,16),COPY(37,298),D(4,4),DL(4,4),
     1DNZR(2,8),ESM(16,16),F(24,298),SSM(298,66),STOR(20,150)
C
      COMMON /TWO/ AL(5),DENS(5),DETA(8),DXI(8),E(5),EPSO(4),
     1FE(16),GC(4),GX(4),POI(5),R(149),SHP(8),SHT(5),TK(5),Z(149)
C
      COMMON /TRE/ AXIALF,DTI,EVERY,FORS1,OMEGA,HTCI,HTCO,
     1POSI,PRES,Q1,Q2,Q3,SHMAX,SMAX,TIME,TIME1,TINIT,TMAXM,TREF,
     2TMAXO,RMAXM,RMAXO,ZMAXM,ZMAXO,Q4,BIG
C
      COMMON /FOR/ NN(40,9)
C
      COMMON /FIV/ NCFI(37),NCFO(37),NNR(37),NRE(37),NNL(37),
     1NPN(37)
      COMMON /SIX/ IBAND,IEXT,INTP,IVEC,JVEC,KK,LOAD,NCFIT,NCFOT,
     1NCHECK,NET,NDF,NGP,NNE,NNLT,NNRT,NPASS,NPROB,NRPPI,NRPPO,
     3NNT,NNRE,NPNT,IPLANE,NBAND
C
      DATA  BRIT/'BRIT'/
C
      PAI = 3.141592653589793
      II = IVEC + 1
C
C
      DO  25  L = 1 , NET
      NNE2 = 2 * NNE
      DO  11  I = 1 , NNE2 , 2
   11 FE(I) = 0.DO
C
C
C
      K = NN(L,9)
```

```
      DUM1 = 2.DO * PAI * DENS(K) * OMEGA**2                            00026310
C                                                                       00026320
      IF  (Q1.EQ.BRIT)  DUM1 = DUM1 / 386.0885827                       00026330
      IF  (Q1.NE.BRIT)  DUM1 = DUM1 / 980.665                           00026340
C                                                                       00026350
      DO  9  N = 1,NGP                                                  00026360
      ETA = GX(N)                                                       00026370
      DO  8  M = 1,NGP                                                  00026380
      XI = GX(M)                                                        00026390
C                                                                       00026400
C                                                                       00026410
      SHP(1) = .25DO * (1.DO-XI) * (1.DO-ETA) * (-XI-ETA-1.DO)          00026420
      SHP(2) = .5DO * (1.DO-XI**2) * (1.DO-ETA)                         00026430
      SHP(3) = .25DO * (1.DO+XI) * (1.DO-ETA) * (XI-ETA-1.DO)           00026440
      SHP(4) = .5DO * (1.DO-ETA**2) * (1.DO+XI)                         00026450
      SHP(5) = .25DO * (1.DO+XI) * (1.DO+ETA) * (XI+ETA-1.DO)           00026460
      SHP(6) = .5DO * (1.DO-XI**2) * (1.DO+ETA)                         00026470
      SHP(7) = .25DO * (1.DO-XI) * (1.DO+ETA) * (-XI+ETA-1.DO)          00026480
      SHP(8) = .5DO * (1.DO-ETA**2) * (1.DO-XI)                         00026490
C                                                                       00026500
C                                                                       00026510
C      DERIVATIVES OF THE SHAPE FUNCTIONS WITH RESPECT TO XI.           00026520
C                                                                       00026530
C                                                                       00026540
      DXI(1)     = .25DO * (1.DO-ETA) * (2.DO*XI+ETA)                   00026550
      DXI(2)     = -XI * (1.DO-ETA)                                     00026560
      DXI(3)     = .25DO * (1.DO-ETA) * (2.DO*XI-ETA)                   00026570
      DXI(4)     = .5DO * (1.DO-ETA**2)                                 00026580
      DXI(5)     = .25DO * (1.DO+ETA) * (2.DO*XI+ETA)                   00026590
      DXI(6)     = -XI * (1.DO+ETA)                                     00026600
      DXI(7)     = .25DO * (1.DO+ETA) * (2.DO*XI-ETA)                   00026610
      DXI(8)     = -.5DO * (1.DO-ETA**2)                                00026620
C                                                                       00026630
C                                                                       00026640
C      DERIVATIVES OF THE SHAPE FUNCTIONS WITH RESPECT TO ETA.          00026650
C                                                                       00026660
C                                                                       00026670
      DETA(1)    = .25DO * (1.DO-XI) * (2.DO*ETA+XI)                    00026680
      DETA(2)    = -.5DO * (1.DO-XI**2)                                 00026690
      DETA(3)    = .25DO * (1.DO+XI) * (2.DO*ETA-XI)                    00026700
      DETA(4)    = -ETA * (1.DO+XI)                                     00026710
      DETA(5)    = .25DO * (1.DO+XI) * (2.DO*ETA+XI)                    00026720
      DETA(6)    = .5DO * (1.DO-XI**2)                                  00026730
      DETA(7)    = .25DO * (1.DO-XI) * (2.DO*ETA-XI)                    00026740
      DETA(8)    = -ETA * (1.DO-XI)                                     00026750
C                                                                       00026760
C                                                                       00026770
C        CONSTRUCTION OF THE JACOBIAN MATRIX.                           00026780
```

```
C
C
      DO   20  I = 1 , 2
      DO   20  J = 1 , 2
   20 AJ(I,J) = 0.D0
C
C
      DO   22  I = 1 , NNE
      AJ(1,1) = AJ(1,1) + DXI(I)   * Z(NN(L,I))
      AJ(1,2) = AJ(1,2) + DXI(I)   * R(NN(L,I))
      AJ(2,1) = AJ(2,1) + DETA(I)  * Z(NN(L,I))
      AJ(2,2) = AJ(2,2) + DETA(I)  * R(NN(L,I))
   22 CONTINUE
C
C
      DET =AJ(1,1) * AJ(2,2) - AJ(1,2) * AJ(2,1)
      DEI =1.0D0/DET
C
C
C     INVERTING THE JACOBIAN AND STORING IN  AJ  AGAIN.
C
C
      DUM = AJ(1,1) * DEI
      AJ(1,1) = AJ(2,2) * DEI
      AJ(1,2) = -AJ(1,2) * DEI
      AJ(2,1) = -AJ(2,1) * DEI
      AJ(2,2) = DUM
C
C
      CR = 0.D0
      DO   2  I = 1,NNE
      M1 = NN(L,I)
    2 CR = CR + SHP(I) * R(M1)
      CR = CR**2
C
C
      DUM = DUM1 * CR * DET * GC(N) * GC(M)
C
C
      DO  17  I = 1 , NNE
      K = 2 * I - 1
      FE(K) = FE(K) + SHP(I) * DUM
   17 CONTINUE
C
C
    8 CONTINUE
    9 CONTINUE
C
```

```
C
C        CONSTRUCTION OF SYSTEM CENTRIFUGAL LOAD VECTOR
C
C
      DO  18  I = 1 , NNE
      I2 = 2 * I - 1
      I1 = NN(L,I) * 2 - 1
      F(II,I1) = F(II,I1) + FE(I2)
   18 CONTINUE
C
C
   25 CONTINUE
C
C
      RETURN
      END
C
```

```
      SUBROUTINE  PRESS
C
C
C        ************************************************************
C                                                                   *
C        SUBROUTINE PRESS EVALUATES THE PRESSURE LOAD VECTOR        *
C        FOR THE SYSTEM WHEN THERE IS ANY PRESSURE (NE.0.DO)        *
C        ACTING ON THE BODY                                         *
C                                                                   *
C        ************************************************************
C
C
      IMPLICIT  REAL * 8 (A-H,O-Z)
C
C
      COMMON /ONE/ AJ(2,2),B(4,16),COPY(37,298),D(4,4),DL(4,4),
     1DNZR(2,8),ESM(16,16),F(24,298),SSM(298,66),STOR(20,150)
C
      COMMON /TWO/ AL(5),DENS(5),DETA(8),DXI(8),E(5),EPSO(4),
     1FE(16),GC(4),GX(4),POI(5),R(149),SHP(8),SHT(5),TK(5),Z(149)
C
      COMMON /TRE/ AXIALF,DTI,EVERY,FORS1,OMEGA,HTCI,HTCO,
     1POSI,PRES,Q1,Q2,Q3,SHMAX,SMAX,TIME,TIME1,TINIT,TMAXM,TREF,
     2TMAXO,RMAXM,RMAXO,ZMAXM,ZMAXO,Q4,BIG
C
      COMMON /FOR/ NN(40,9)
C
      COMMON /FIV/ NCFI(37),NCFO(37),NNR(37),NRE(37),NNL(37),
     1NPN(37)
      COMMON /SIX/ IBAND,IEXT,INTP,IVEC,JVEC,KK,LOAD,NCFIT,NCFOT,
     1NCHECK,NET,NDF,NGP,NNE,NNLT,NNRT,NPASS,NPROB,NRPPI,NRPPO,
     3NNT,NNRE,NPNT,IPLANE,NBAND
C
C
      PAI = 3.141592653589793
      L = 1
      IF  (OMEGA.NE.0.DO)  L = L + 1
      II = IVEC + L
C
C
C
C
      NEUP = (NPNT-1) / 2
C
C        NEUP  IS THE NUMBER OF ELEMENTS UNDER PRESSURE
C
```

```
      FACT = 2.DO * PAI * PRES
C
C
      DO  10  L = 1 , NEUP
      K = 2 * L - 1
      DO  2  I = 1 , NNE
    2 FE(I) = 0.DO
C
C
      DO  9  N = 1 , NGP
      XI = GX(N)
C
C
      SHP(1) = -.5DO * (1.DO - XI) * XI
      SHP(2) = 1.DO - XI**2
      SHP(3) =  .5DO * (1.DO + XI) * XI
C
C
      DXI(1) = XI - .5DO
      DXI(2) = -2.DO * XI
      DXI(3) = XI + .5DO
C
C
      RR = 0.DO
      DNZ = 0.DO
      DNR = 0.DO
C
C
      DO  3  I = 1 , 3
      J = NPN(K+I-1)
      RR  = RR  + SHP(I) * R(J)
      DNZ = DNZ + DXI(I) * Z(J)
      DNR = DNR + DXI(I) * R(J)
    3 CONTINUE
C
C
      DO  4  I = 1 , 3
      J1 = 2 * I - 1
      J2 = 2 * I
      FE(J1) = FE(J1) + RR * DNZ * SHP(I) * GC(N)
      FE(J2) = FE(J2) - RR * DNR * SHP(I) * GC(N)
    4 CONTINUE
    9 CONTINUE
C
C
C         CONSTRUCTION OF SYSTEM PRESSURE LOAD VECTOR
C
C
```

```
      DO 5 I=1,3
      J1=2*I-1
      J2=2*I
      J =2 * NPN(K+I-1) - 1
      F(II,J) = F(II,J) + FE (J1)* FACT
      F(II,J+1) = F(II,J+1) + FE(J2) * FACT
    5 CONTINUE
   10 CONTINUE
C
C
      RETURN
      END
```

```
      SUBROUTINE  DISPL
C
C
C        ************************************************************
C                                                                   *
C        THIS SUBROUTINE SOLVES FOR DISPLACEMENTS,ONCE GIVEN        *
C        THE DECOMPOSED SYSTEM STIFFNESS MATRIX AND THE LOAD        *
C        VECTORS.     THE LOAD VECTORS ARE REPLACED BY THE          *
C        SOLUTIONS  'THE DISPLACEMENTS' .                           *
C                                                                   *
C        ************************************************************
C
C
      IMPLICIT  REAL * 8 (A-H,O-Z)
C
      COMMON /ONE/ AJ(2,2),B(4,16),COPY(37,298),D(4,4),DL(4,4),
     1DNZR(2,8),ESM(16,16),F(24,298),SSM(298,66),STOR(20,150)
C
      COMMON /TWO/ AL(5),DENS(5),DETA(8),DXI(8),E(5),EPSO(4),
     1FE(16),GC(4),GX(4),POI(5),R(149),SHP(8),SHT(5),TK(5),Z(149)
C
      COMMON /TRE/ AXIALF,DTI,EVERY,FORS1,OMEGA,HTCI,HTCO,
     1POSI,PRES,Q1,Q2,Q3,SHMAX,SMAX,TIME,TIME1,TINIT,TMAXM,TREF,
     2TMAXO,RMAXM,RMAXO,ZMAXM,ZMAXO,Q4,BIG
C
      COMMON /FOR/ NN(40,9)
C
      COMMON /FIV/ NCFI(37),NCFO(37),NNR(37),NRE(37),NNL(37),
     1NPN(37)
      COMMON /SIX/ IBAND,IEXT,INTP,IVEC,JVEC,KK,LOAD,NCFIT,NCFOT,
     1NCHECK,NET,NDF,NGP,NNE,NNLT,NNRT,NPASS,NPROB,NRPPI,NRPPO,
     3NNT,NNRE,NPNT,IPLANE,NBAND
C
C
      ZERO = 0.D0
      II = IVEC + JVEC + 1
      IF (IPLANE.EQ.0)  II = II - 1
      DO  500  LL = 1 , II
C
C
C
      DO  200  I = 1 , NDF
      F(LL,I) = F(LL,I) / SSM(I,1)
      DO  100  J = 2 , IBAND
      L = I + J - 1
      IF  (L. GT .NDF)  GO  TO  200
```

```
      DUM = SSM(I,J)
  100 F(LL,L) = F(LL,L) - DUM * F(LL,I)
  200 CONTINUE
C
C
C
      F(LL,NDF) = F(LL,NDF) / SSM(NDF,1)
      DO  400  L = 2 , NDF
      K = NDF - L + 1
      SUM = ZERO
      DO  300  J = 2 , IBAND
      M = J + K - 1
      IF  (NDF. LT .M)  GO  TO  400
  300 SUM = SUM + SSM(K,J) * F(LL,M)
  400 F(LL,K) = (F(LL,K) - SUM) / SSM(K,1)
C
C
C
  500 CONTINUE
C
C
      IF (IPLANE.EQ.0)  GO  TO  20
C
C
C        CORRECTING THE DISPLACEMENTS
C
C
      FORS1 = 0.D0
      DO  7  I = 1 , NNRE
      DUM = 0.D0
      DO  6  J = 1 , NDF
    6 DUM = DUM + COPY(I,J) * F(II,J)
    7 FORS1 = FORS1 + DUM
C
C
      IF  (JVEC.EQ.0)  GO  TO  10
C
      DO  8  I = 1 , JVEC
      SUM = 0.D0
      DO  9  J = 1 , NNRE
      K = 2 * NRE(J)
      SUM = SUM + COPY(J,K) * F(IVEC+I,K) * BIG
    9 CONTINUE
C
      AXIALF = AXIALF + SUM
C
    8 CONTINUE
   10 CONTINUE
```

```
C
C
      DO  11  I = 1 , IVEC
      SUM = 0.D0
      DO  12  J = 1 , NNRE
      K = 2 * NRE(J)
      SUM = SUM + COPY(J,K) * F(I,K) * BIG
   12 CONTINUE
C
      FACT = (AXIALF+ SUM)/FORS1
C
      DO  28  L = 1 , NDF
      F(I,L) = F(I,L) + FACT * F(II,L)
   28 CONTINUE
   11 CONTINUE
C
C
C         USING THE PRINCIPLE OF SUPERPOSITION
C
C
   20 M = 0
      IF  (LOAD. NE . 0)   M = M + 1
      IF  (PRES .NE.0.DO)   M = M + 1
      IF  (OMEGA.NE.0.DO)   M = M + 1
      IF  (M. EQ . 0)   GO  TO  40
      DO  41  I = 1 , IVEC
      DO  41  J = 1 , NDF
      DO  41  K = 1 , M
      F(I,J) = F(I,J) + F(IVEC+K,J)
   41 CONTINUE
C
C
   40 CONTINUE
      RETURN
      END
```

```
C
      SUBROUTINE  STRESS
C
C
C        ************************************************************
C                                                                   *
C        AT THE TWO GAUSS POINTS ON THE TWO BOUNDARIES OF           *
C        EACH ELEMENT (WHERE ETA=+1 AND ETA=-1) STRESSES            *
C        ARE CALCULATED IN THIS SUBROUTINE, AND ALSO THE            *
C        MAXIMUM 'MEAN AND OCTAHEDRAL' SHEARING STRESSES            *
C        ARE EVALUATED AND THEN THE TIME AND LOCATION OF            *
C        THE MAXIMUM STRESSES ARE DISTINGUISHED HERE.               *
C                                                                   *
C        ************************************************************
C
C
      IMPLICIT  REAL * 8 (A-H,O-Z)
C
      COMMON /ONE/ AJ(2,2),B(4,16),COPY(37,298),D(4,4),DL(4,4),
     1DNZR(2,8),ESM(16,16),F(24,298),SSM(298,66),STOR(20,150)
C
      COMMON /TWO/ AL(5),DENS(5),DETA(8),DXI(8),E(5),EPSO(4),
     1FE(16),GC(4),GX(4),POI(5),R(149),SHP(8),SHT(5),TK(5),Z(149)
C
      COMMON /TRE/ AXIALF,DTI,EVERY,FORS1,OMEGA,HTCI,HTCO,
     1POSI,PRES,Q1,Q2,Q3,SHMAX,SMAX,TIME,TIME1,TINIT,TMAXM,TREF,
     2TMAXO,RMAXM,RMAXO,ZMAXM,ZMAXO,Q4,BIG
C
      COMMON /FOR/ NN(40,9)
C
      COMMON /FIV/ NCFI(37),NCFO(37),NNR(37),NRE(37),NNL(37),
     1NPN(37)
C
      COMMON /SIX/ IBAND,IEXT,INTP,IVEC,JVEC,KK,LOAD,NCFIT,NCFOT,
     1NCHECK,NET,NDF,NGP,NNE,NNLT,NNRT,NPASS,NPROB,NRPPI,NRPPO,
     3NNT,NNRE,NPNT,IPLANE,NBAND
C
      DIMENSION  EDISP(16),SIG(4),EPS(4)
C
      GX(1) = -.577350269189626
      GX(2)=-GX(1)
      GC(1)=1.0D0
      GC(2)=GC(1)
      PAI = 3.14159265358979З
      NNE2 = 2 * NNE
      DO  25  LL= 1 ,NET
```

```
C
      WRITE  (6,200)  LL
  200 FORMAT  (1H1,//,40X,'STRESSES AND TEMPERATURES IN ELEMENT',
     1I3,/,40X,39('*'),/,40X,39('*'),/)
C
      L = LL
C
      IF  (L. EQ .1 )  GO  TO  1
      IF  (NN(L,9).EQ.NN(L-1,9))  GO  TO  3
C
    1 K = NN(L,9)
      EL = 1.D0 - POI(K)
      FACT = (E(K)*EL)/((1.D0+POI(K))*(1.D0-2.D0*POI(K)))
      EL = POI(K) / EL
C
      D(1,1) = FACT
      D(1,2) = EL * FACT
      D(1,3) = D(1,2)
      D(1,4) = 0.D0
      D(2,2) = FACT
      D(2,3) = D(1,2)
      D(2,4) = 0.D0
      D(3,3) = FACT
C

      D(3,4) = 0.D0
      D(4,4) = E(K)/(2.D0*(1.D0+POI(K)))
C
C
      DO  3  I = 1,4
      DO  3  J = I,4
      D(J,I) = D(I,J)
C
    3 CONTINUE
C
C
C
C
      ETA = -1.D0
      DO  9  N = 1 , 2
      ETA = - ETA
      DO  13  M = 1 , 2
      XI = GX(M)
C
C
      DO  7  I = 1,4
      DO  7  J = 1,NNE2
    7 B(I,J) = 0.D0
```

```
C
C
C
      SHP(1) = .25DO * (1.DO-XI) * (1.DO-ETA) * (-XI-ETA-1.DO)
      SHP(2) = .5DO * (1.DO-XI**2) * (1.DO-ETA)
      SHP(3) = .25DO * (1.DO+XI) * (1.DO-ETA) * (XI-ETA-1.DO)
      SHP(4) = .5DO * (1.DO-ETA**2) * (1.DO+XI)
      SHP(5) = .25DO * (1.DO+XI) * (1.DO+ETA) * (XI+ETA-1.DO)
      SHP(6) = .5DO * (1.DO-XI**2) * (1.DO+ETA)
      SHP(7) = .25DO * (1.DO-XI) * (1.DO+ETA) * (-XI+ETA-1.DO)
      SHP(8) = .5DO * (1.DO-ETA**2) * (1.DO-XI)
C
C
C      DERIVATIVES OF THE SHAPE FUNCTIONS WITH RESPECT TO XI.
C
C
      DXI(1)      = .25DO * (1.DO-ETA) * (2.DO*XI+ETA)
      DXI(2)      = -XI * (1.DO-ETA)
      DXI(3)      = .25DO * (1.DO-ETA) * (2.DO*XI-ETA)
      DXI(4)      = .5DO * (1.DO-ETA**2)
      DXI(5)      = .25DO * (1.DO+ETA) * (2.DO*XI+ETA)
      DXI(6)      = -XI * (1.DO+ETA)
      DXI(7)      = .25DO * (1.DO+ETA) * (2.DO*XI-ETA)
      DXI(8)      = -.5DO * (1.DO-ETA**2)
C
C
C      DERIVATIVES OF THE SHAPE FUNCTIONS WITH RESPECT TO ETA.
C
C
      DETA(1)     = .25DO * (1.DO-XI) * (2.DO*ETA+XI)
      DETA(2)     = -.5DO * (1.DO-XI**2)
      DETA(3)     = .25DO * (1.DO+XI) * (2.DO*ETA-XI)
      DETA(4)     = -ETA * (1.DO+XI)
      DETA(5)     = .25DO * (1.DO+XI) * (2.DO*ETA+XI)
      DETA(6)     = .5DO * (1.DO-XI**2)
      DETA(7)     = .25DO * (1.DO-XI) * (2.DO*ETA-XI)
      DETA(8)     = -ETA * (1.DO-XI)
C
C
C        CONSTRUCTION OF THE JACOBIAN MATRIX.
C
C
      DO  20  I = 1 , 2
      DO  20  J = 1 , 2
   20 AJ(I,J) = 0.DO
C
C
      DO  22  I = 1 , NNE
```

```
      AJ(1,1) = AJ(1,1) + DXI(I)  * Z(NN(L,I))
      AJ(1,2) = AJ(1,2) + DXI(I)  * R(NN(L,I))
      AJ(2,1) = AJ(2,1) + DETA(I) * Z(NN(L,I))
      AJ(2,2) = AJ(2,2) + DETA(I) * R(NN(L,I))
   22 CONTINUE
C
C
      DET =AJ(1,1) * AJ(2,2) - AJ(1,2) * AJ(2,1)
      DEI =1.0D0/DET
C
C
C     INVERTING THE JACOBIAN AND STORING IN  AJ  AGAIN.
C
C
      DUM = AJ(1,1) * DEI
      AJ(1,1) = AJ(2,2) * DEI
      AJ(1,2) = -AJ(1,2) * DEI
      AJ(2,1) = -AJ(2,1) * DEI
      AJ(2,2) = DUM
C
C
C
      CR = 0.D0
      DO  2  I = 1,NNE
      M1 = NN(L,I)
    2 CR = CR + SHP(I) * R(M1)
C
C
C
C     EVALUATION OF THE B MATRIX
C
C
      DO  15  I = 1, NNE
      K1 = 2 * I - 1
      K2 = K1 + 1
      B(1,K2) = AJ(1,1) * DXI(I) + AJ(1,2) * DETA(I)
      B(2,K1) = AJ(2,1) * DXI(I) + AJ(2,2) * DETA(I)
      B(3,K1) = SHP(I) / CR
      B(4,K1) = B(1,K2)
      B(4,K2) = B(2,K1)
   15 CONTINUE
C
C        POSITION OF GAUSS POINT
C
C
      RPOS = 0.D0
      ZPOS = 0.D0
      DO  8  I = 1 , NNE
```

```
      RPOS = RPOS + SHP(I) * R(NN(L,I))
    8 ZPOS = ZPOS + SHP(I) * Z(NN(L,I))
C
C
      WRITE  (6,201)  XI , ETA , RPOS , ZPOS
  201 FORMAT  (//,40X,'AT LOCATION XI=',1PD14.7,'  AND ETA=',
     11PD14.7,/,53X,'R=',1PD14.7,'  AND  Z =',1PD14.7,//,4X,
     2'TIME',8X,'SIGMA  Z',9X,'SIGMA  R',8X,'SIGMA THETA',7X,
     3'TAU  R Z',7X,'TEMPERATURE',6X,'MEAN STRESS',7X,'OCTA ',
     4'SHEAR',//)
C
      DO  30  KK = 1 , IVEC
C
C
C        EVALUATION OF THE DISPLACEMENT OF ELEMENT L
C
C
      DO  4  I = 1 , NNE
      I2 = I * 2
      I1 = I2 - 1
      J2 = NN(L,I) * 2
      J1 = J2 - 1
      EDISP(I1) = F(KK,J1)
    4 EDISP(I2) = F(KK,J2)
C
C
C        EVALUATING TEMPERATURE AT GAUSS POINT
C
C
      TGP = 0.D0
      DO  31  I = 1 , NNE
      M1 = NN(L,I)
   31 TGP = TGP + SHP(I) * STOR(KK,M1)
C
      TGP = TGP -TREF
      TGP = AL(NN(L ,9)) * TGP
C
C
C        EVALUATION OF STRAIN VECTOR FOR A GAUSS POINT ON
C        THE BOUNDARY OF ELEMENT L .
C
C
      DO  5  I = 1 , 4
      G = 0.D0
      DO  5  J = 1 , NNE2
      G = G + B(I,J) * EDISP(J)
    5 EPS(I) = G
C
```

```
C           NOTE: INITIAL STRAIN (4) IS ZERO AND IS NOT
C           BEING SUBTRACTED FROM EPS(4)
C
       DO  6   I = 1 , 3
     6 EPS(I) = EPS(I) - TGP
C
C
       DO  10  I = 1 , 4
       G = 0.D0
       DO  10  J = 1 , 4
       G = G + D(I,J) * EPS(J)
    10 SIG(I) = G
C
C
C
C
C           SMEAN   IS THE MEAN STRESS
C           TMAXM   IS THE TIME WHEN MAX. MEAN STRESS OCCURS.
C           OCTA    IS THE OCTAHEDRAL SHEAR STRESS
C           TMAXO   IS THE TIME WHEN MAX. OCTAHEDRAL  SHEAR
C           STRESS OCCURS.
C
       SMEAN = (SIG(1)+SIG(2)+SIG(3))/3.D0
C
C
C
C           CALCULATION OF OCTAHEDRAL SHEAR STRESS
C
       OCTA = 0.D0
       OCTA = OCTA + (SIG(1)-SIG(2))**2 + (SIG(2)-SIG(3))**2
       OCTA = (OCTA + (SIG(3)-SIG(1))**2) / 9.D0
       OCTA = OCTA + (2.D0/3.D0) * SIG(4)**2
       OCTA = DSQRT(OCTA)
C
C
C
       TGP = TGP / AL(NN(L,9)) + TREF
       TIME = STOR(KK,NNT+1)
C
       WRITE  (6,202)  (TIME,(SIG(I),I=1,4),TGP,SMEAN,OCTA)
   202 FORMAT  (3X,F7.2,7(3X,1PD14.6))
C
C
C
C           FINDING MAX. MEAN STRESS
C
       SMEAN = DABS(SMEAN)
       IF  (SMEAN.GT.SMAX)  GO  TO  27
```

```
      GO  TO  29
   27 SMAX = SMEAN
      TMAXM  =  STOR(KK,NNT+1)
      RMAXM  =  RPOS
      ZMAXM  =  ZPOS
   29 CONTINUE
C
C
C        FINDING MAX. OCTAHEDRAL  SHEAR STRESS.
C
C
      IF  (OCTA.GT.SHMAX)  GO  TO  28
      GO  TO  30
   28 SHMAX  =  OCTA
      TMAXO  =  STOR(KK,NNT+1)
      RMAXO  =  RPOS
      ZMAXO  =  ZPOS
C
C
C
   30 CONTINUE
   13 CONTINUE
    9 CONTINUE
   25 CONTINUE
C
C
      RETURN
      END
```

## APPENDIX C

## USER'S MANUAL

In this section the procedure for the use of the present computer program is described. It is intended that a person with minimum familiarity with the details of the finite element method and computer programming be able to use this program.

The steps below are in order and the user is advised to follow them carefully.

For finding the thermal stresses in an axisymmetric body under axisymmetric loading, the user has the choice of using either the Metric or British system of units. The program will handle both systems and the necessary conversions are made automatically within the program. However, the units used in each system must be consistent and they should be as listed below in Table V.

Now for preparation of the data input for a given axisymmetric geometry with prescribed thermal and structural boundary conditions we go through the details with a simple example.

Step 1:

Draw the longitudinal cross-section of the body to scale. Identify the cylindrical coordinates R and Z, with the origin of the Z axis on the most left-hand point of the

TABLE V

UNITS FOR INPUT DATA

Note: an input card specifies whether British or Metric data is being used.

| Quantity | British Unit Syst. | Metric Syst. |
|---|---|---|
| Coordinates | in. | cm |
| Time | sec. | sec. |
| Temperature | °F | °C |
| Velocity | ft./sec. | m/sec. |
| Axial force✓ | lbf | kg |
| Pressure | lbf/in$^2$ | kg/cm$^2$ |
| Density | lbm/ft$^3$ | gm/cm$^3$ |
| Specific heat | Btu/lbm°F | cal/kg°C |
| Mod. of elasticity | lbf/in$^2$ | kg/mm$^2$ |
| Coef. of thermal exp. | 1/°F | 1/°C |
| Thermal conductivity | Btu/hr.ft.°F | cal/sec.cm.°C |
| Heat transfer coef. | Btu/hr.ft.$^2$°F | cal/sec.cm.$^2$°C |
| Load vector | lbf | kg |
| Rotational speed | Revolutions/min. | Revolutions/min. |

✓lbf is pound force and lbm is pound mass.

body. The entire cross section must lie in the first quadrant of the coordinate plane.

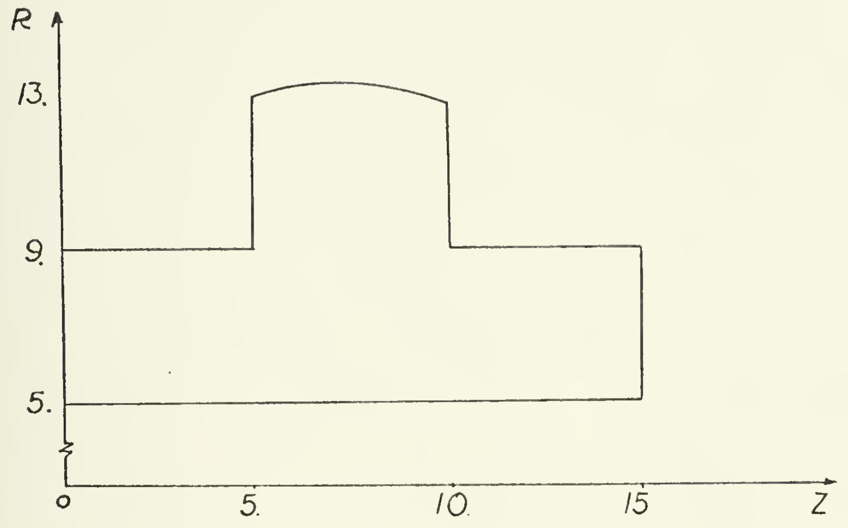


Figure 14. Longitudinal Cross Section.

Step 2:

Subdivide the cross section into a maximum of 40 quadrilateral elements. This subdivision is extremely important and we should provide smaller elements for the parts of the cross section where we expect the largest stresses or temperature gradient. If the body is made from several materials, elements should be chosen so that each element contains only one material. Any two adjacent elements must share one complete side of the quadrilateral. Number the elements, starting from 1, in any arbitrary manner (see Fig. 15).

Step 3:

Since eight-noded elements are used in the program, identify these nodal points around the boundary of each element. Four nodes will be at the corners and the other four will be

at the mid-points of the sides. Number sequentially these nodes starting from 1 and increasing in the direction where the number of elements is the least. (The numbers thus assigned are called global node numbers.) For clarity of this step assume a cross-section as in Figure 15 such that the maximum number of elements in one direction is less than the maximum number of elements in other direction. (In Fig. 15 we have maxima of 2 and 3 elements in R and Z directions respectively.) Thus we number the nodes beginning in the R direction. See Fig. 16. This method will give the minimum band width of the stiffness matrix and will save computer execution time.

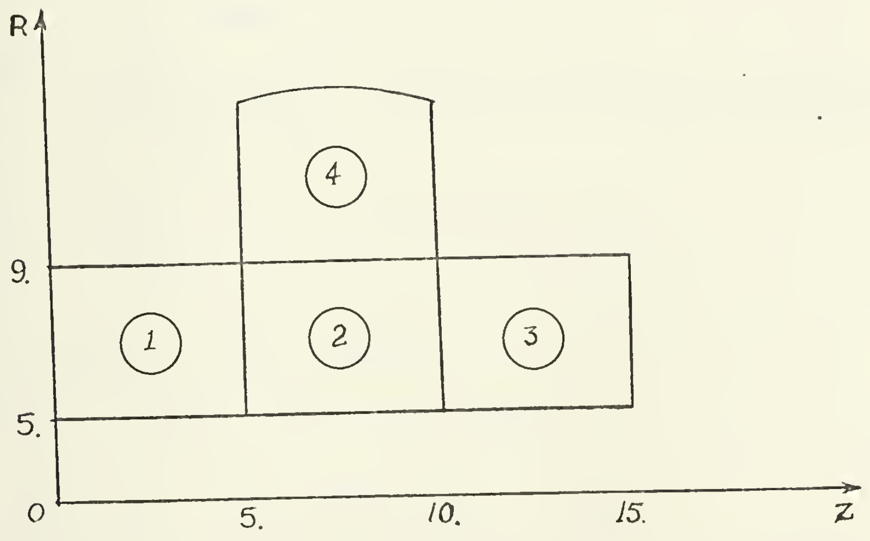


Figure 15. Subdivision Inot Elements.

Step 4:

At this point we are ready to prepare the first data card. Input quantities are:

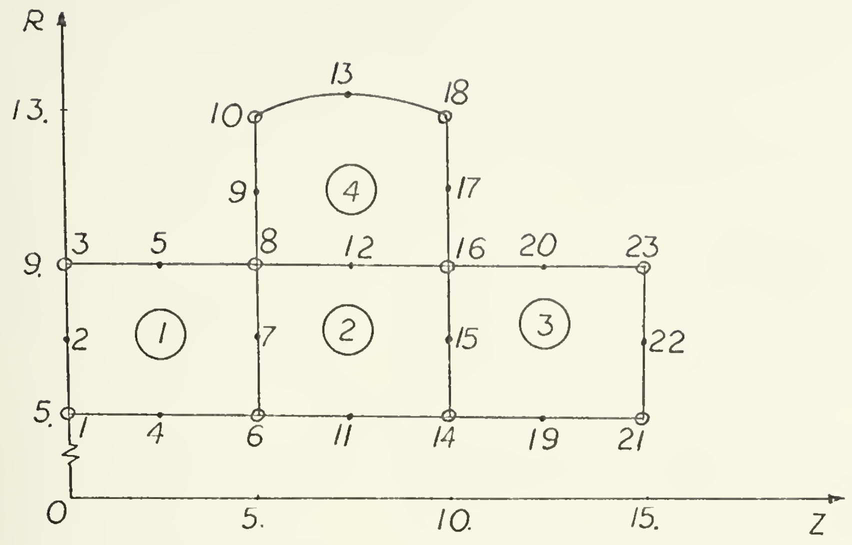


Figure 16. Element and Node Numbering.

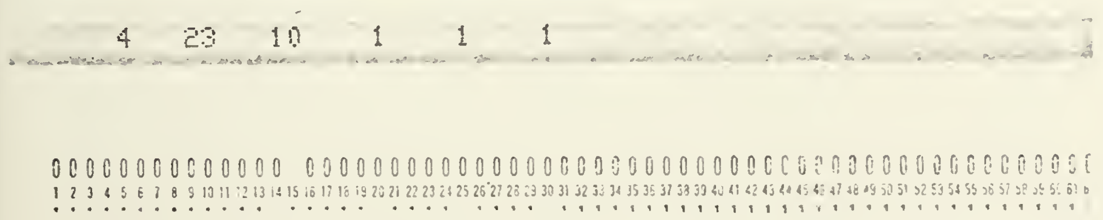
| | |
|---|---|
| NET | The total number of elements |
| NNT | Total number of nodes |
| NCN | Total number of corner nodes |
| NØFN | Total number of mid-side nodes not lying on the straight line joining the corner nodes (number of "Off" nodes) |
| NMAT | Number of different materials |
| NPRØB | Number of problems to be solved |

For the present we take NPRØB = 1.

Prepare a single card that reads

NET, NNT, NCN, NØFN, NMAT, NPRØB

with the format (8I5). For our example, if all the elements are from the same material this data card reads:



```
    4   23   10    1    1    1
```

On this card, or any succeeding one, if an input quantity (e.g., NØFN) is zero, the corresponding field may have a 0 or be left blank, unless otherwise stated.

Step 5:

For each corner node a card should be punched which reads: The node number I, the R coordinate and the Z coordinate of the node. The format is (I5, 2F10.5).

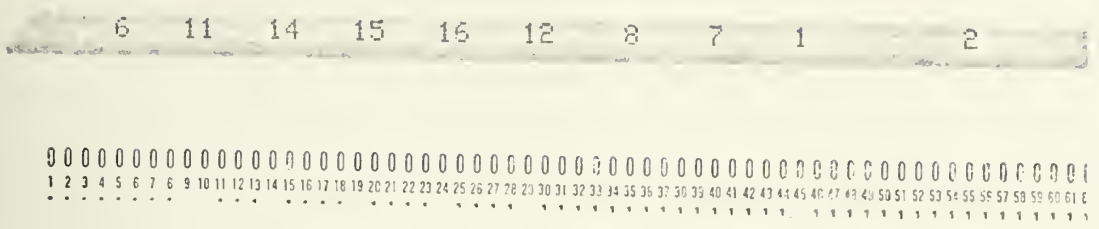
It is not necessary to sort these cards in the order of increasing or decreasing corner node numbers, they can be put together in any arbitrary order. A typical card is illustrated in step 7. There must be as many punched cards as the number of corner nodes (NCN).

Step 6:

In this step we read in the connectivity array, i.e., for each element we prepare one card which gives the global node numbers and the material identification number.

For each element, start from the lower left-hand node and move in the counterclockwise direction within that element and punch the global node numbers in order. The format is (9I5) where the last I5 is the material identification number.

It is very important that the cards prepared for this step be put together in order of elements, i.e., the first card for element 1, the second card for element 2, etc. For ease of sorting the connectivity cards the element number may be punched after column 50. As an example, for element 2 of Fig. 16 the connectivity card should read:



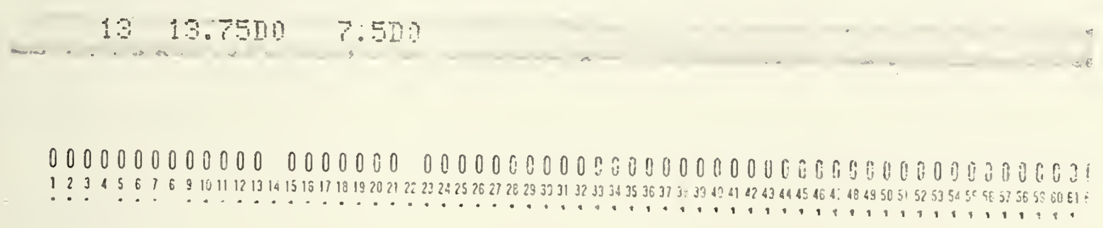
```
    6   11   14   15   16   12    8    7    1         2
```

where 1 identifies the type of material in this element and 2 in column 55 stands for the element number.

Step 7:

If each side of every element is straight, NØFN = 0 and this step is omitted.

For each mid-side node that is on a curved element edge a card is prepared with format (I5,2F10.5) to read the global node number and the R and Z coordinates. Here, as in step 5, these cards may be put together in any arbitrary order. For the case of Fig.16, there will be only one card to be punched and it would read:



```
   13  13.75D0   7.5D0
```

To assure in all cases that the double precision capability of the program is not compromised by less precise floating point inputs, it is recommended that all such inputs be made in D-FORMAT. The F-FORMAT instructions merely allocate input card fields.

Step 8:

The properties of the different materials are specified in this step. For each material a card must be prepared that gives the modulus of elasticity (E), coefficient of thermal expansion (AL), Poisson's ratio (PØI), thermal conductivity (TK), density (DENS) and specific heat (SHT). I.e., read:

```
        E, AL, PØI, TK, DENS, SHT
```

with the format (2D12.4, 4F8.3). The first card will be for material number 1, the second card for material number 2, etc.

Step 9:

For this step a single card must be punched, starting in column one, which reads the word BRITISH or METRIC in accordance with the system of units used.

This completes the input of geometric and material property data.

Step 10:

For a transient temperature problem only (no stress calculations) leave a blank card for this step and proceed directly to step 16.

For a stress problem or thermal stress problem, in this step we specify the type of problem and the structural boundary conditions.

The following quantities, when pertinent, are to be specified.

| | |
|---|---|
| ØMEGA | The speed of rotation, about the Z axis, in revolutions per minute |
| PRES | The external pressure applied to a boundary segment |
| IEXT | The number of known temperature vectors for which evaluation of thermal stresses is desired |
| LØAD | The indication for an additional known load vector. If there is one, LØAD = 1, otherwise LØAD = 0 |
| NNLT | Total number of nodes fixed in the longitudinal direction |
| NNRT | Total number of nodes fixed in the radial direction |
| IPLANE | An indication for the plane-end boundary condition. If desired IPLANE = 1 |

If any of these items is not applicable, the corresponding field is left blank.

Now we have to punch a single card for this step which reads:

ØMEGA, PRES, IEXT, LØAD, NNLT, NNRT, IPLANE

with format (2F10.4, 5I5).

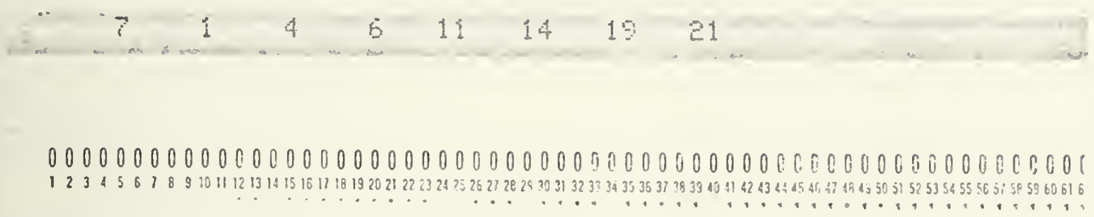
Step 11:

If there is no pressure loading (PRES=0), omit this step. For non-zero pressure, the total number of nodes on pressure loading boundary segment (NPNT) and the global node numbers of these (pressure) nodes (NPN(I)) must be specified here. Only one segment is permitted.
We read in:

NPNT, (NPN(I),I=1,NPNT)

with the format (12I5).

In our example, Fig. 16, if there is pressure inside the body, then the data card for this step will be:



```
   7    1    4    6   11   14   19   21
```

```
0000000000000000000000000000000000000000000000000000000000000
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 6
```

<u>Step 12</u>:

If there is no additional load vector (LØAD = 0), omit this step.

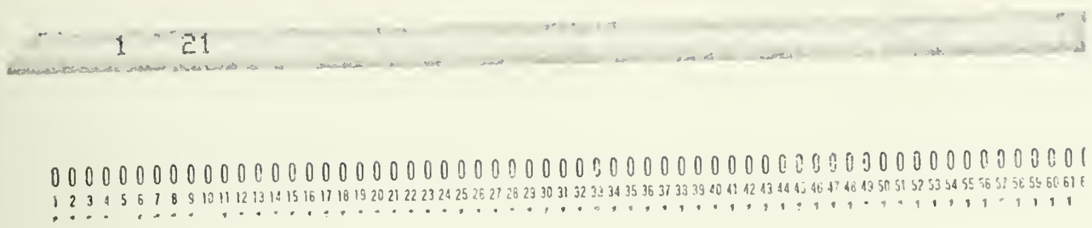
For LØAD = 1 read in the additional load vector with the format (6F10.4). The components of the load vector are arranged in the order: R component at node 1, Z component at node 1, R component at node 2, etc. I.e., READ (F(I),I=1,NDF), where NDF (the number of degrees of freedom) is equal to two times the total number of nodes (2*NNT).

<u>Step 13</u>:

In any stress problem, or thermal stress problem, there must be at least one node constrained against longitudinal motion, i.e., NNLT > 0, so we must specify here which nodes are to be constrained against longitudinal motion. These global node numbers NNL(I) are read in with the format (10I5), i.e.,

READ: (NN(I),I=1,NNLT)

In our example, if global nodes 1 and 21 are fixed in the Z direction, we have a card that reads:



```
    1   21
```

Step 14:

If there are no nodes fixed in the radial direction (NNRT = 0), omit this step.

For NNRT > 0 we must read in the global numbers (NNR(I)) of those nodes which are to be fixed against radial motion. The format is (10I5), i.e.,

```
        READ:  (NNR(I),I=1,NNRT)
```

If node 1 and 21 of our example are also fixed in the R direction, then the card for this step would read exactly as the one in Step 13.

Step 15:

If there is no plane-end boundary condition (IPLANE = 0), omit this step.

For the case of end-planes-remain-plane boundary condition, i.e., (IPLANE = 1), we have to specify the total number of nodes of the right-hand end (NNRE) and the global node numbers of this end (NNR(I)).

We read in:

```
        NNRE, (NNR(I),I=1,NNRE)
```

with the format (10I5).

For the case of Fig. 16, if it is desired to have end-planes-remain-plane boundary condition then the data for this step will be:

Step 16:

In this step we begin the specification of thermal boundary conditions. Additional details are prescribed in Steps 19 through 24.

Thermal boundary conditions are imposed along discrete segments of the boundary. Each such segment must begin and end at a corner node of a boundary element. We consider separately the imposition of the various kinds of thermal boundary conditions.

(a) Convection

Since the fluid temperature for convection is determined from a prescribed temperature-time history at entry section and a specified flow velocity (see Appendix D, Section 5), it is necessary that the terms "inside," i.e., adjacent to the symmetry axis, and "outside" have their ordinary meanings. Thus on the inside surface the convection boundary condition may be applied to a single (continuous) segment. A similar prescription may be employed for the outside portion. The entry section used for flow calculations is at the upstream end of corresponding segment.

(b) Constant temperature

To specify constant temperature on a portion of the boundary, the designator "inside" or "outside" may be used. Such a constant temperature portion may consist of several

discrete segments. If two different constant temperature portions are prescribed, one may be designated "inside" and the other "outside."

(c) Combinations of convection and constant temperature

The convection and constant temperature conditions can be used together, but the portion designated "inside" must have only one of these conditions prescribed and the same restriction applies to the "outside."

(d) Insulated

All portions of the boundary not included in the segments specifically identified as "inside" and "outside" are considered insulated.

The following items, when pertinent, are to be given as specified below:

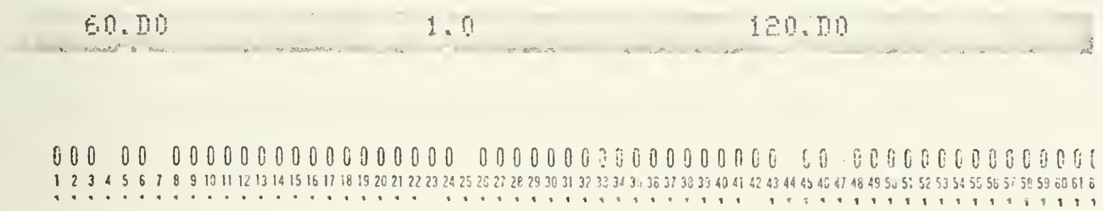
| | |
|---|---|
| TINIT | The constant initial body temperature |
| $Q2=0.$ | For insulated boundary condition outside |
| $Q2 > 0.$ | For convection or constant temperature boundary condition outside |
| $Q3=0.$ | For insulated boundary condition inside |
| $Q3 > 0.$ | For convection or constant temperature boundary condition inside |
| $Q4 < 0.$ | If solving stress problem only |
| $Q4 > 0.$ | If solving temperature problem only |
| $Q4=0.$ | If solving thermal stress problem |
| AXIALF | The axial force in the Z direction. As usual, + (plus) for tension and - (minus) for compression. |

So, we read:

TINIT, Q2, Q3, Q4, AXIALF

with the format (5F10.4).

In case of Fig. 16, if the initial solid temperature is 60° and we have insulated outside and convection boundary condition inside with 120 kg axial compressive force, then the card for this step will be:



```
   60.D0                1.0                  120.D0
```

where the blanks left in the columns 31 through 40 indicate that we want to solve a thermal stress problem.

Step 17:

No action is taken in this step - we merely choose between proceeding to step 18 or jumping to step 26.

If there are no known temperature vectors for which evaluation of thermal stresses is desired, proceed to step 18.

For IEXT > 0, i.e., when some known temperature vectors are to be entered for thermal stress analysis alone or combined with some other loadings, proceed directly to step 26.

Step 18:

If (Q4 < 0), i.e., we are to solve only a stress problem, proceed directly to step 27.

Step 19:

If there is an insulated boundary condition outside (Q2 = 0), omit the following steps and start from step 22.

For the case of a convection or constant temperature boundary condition outside (Q2 > 0.), we have to specify the outside heat transfer coefficient (HTCØ)(for constant temperature HTCØ = $10^{20}$.), the constant outside initial fluid temperature (TEMPØ) which may be equal to the solid's initial temperature, the total number of nodes in contact with fluid outside (NCFØT), and the number of temperature ramps for outside flow (NRAMPØ). We read:

```
HTCØ, TEMPØ, NCFØT, NRAMPØ
```

with the format (D16.4,F10.4,2I5). See example in step 22.

Step 20:

Here we read in the global node numbers of the nodes in contact with outside fluid (NCFØ(I)). The sequence of these node numbers must be in the direction of flow velocity. Read:

```
(NCFØ(I),I=1,NCFØT)
```

with the format (12I5) (see example in step 23).

Step 21:

For each ramp of outside flow we read in the time when the ramp starts BDRYØ(I,1), the initial temperature of the ramp BDRYØ(I,2), the final temperature of the ramp (specify only if it differs from the initial temperature of the next

ramp) BDRYØ(I,3) and the velocity of the fluid for this ramp BDRYØ(I,4), with the format (4F10.4) so we read:

```
        ((BDRYØ(I,J),J=1,4),I=1,NRAMPØ).
```

The number of cards prepared at this step is equal to the number of ramps. See example in step 24.

<u>Step 22</u>:

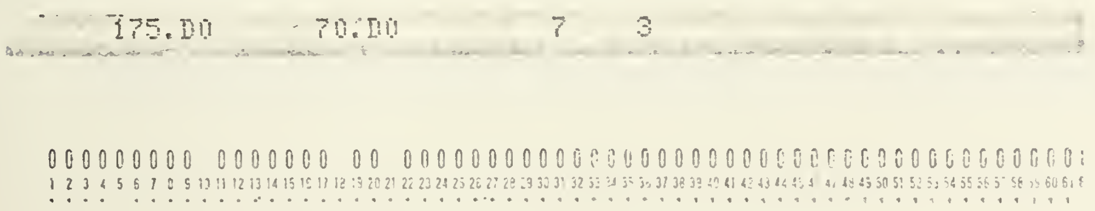
If there is an insulated boundary condition inside (Q3=0), proceed directly to step 25.

For the case of convection or constant temperature boundary condition inside, $Q3 > 0$. We have to specify the inside heat transfer coefficient (HTCI) (for constant inside temperature $HTCI=10.^{20}$), the constant inside initial fluid temperature (TEMPI), which may be equal to the solid's initial temperature, total number of nodes in contact with fluid inside (NCFIT) and the number of ramps of the inside flow (NRAMPI). We read:

```
        HTCI, TEMPI, NCFIT, NRAMPI
```

with the format (D16.4, F10.4,2I5).

For our example assume the time variation of entry temperature for the inside flow to be as in Fig. 17. The heat transfer coefficient is 175.0 with initial inside fluid temperature 70.0. The card for this step is:



```
      175.D0     70.D0        7    3
```

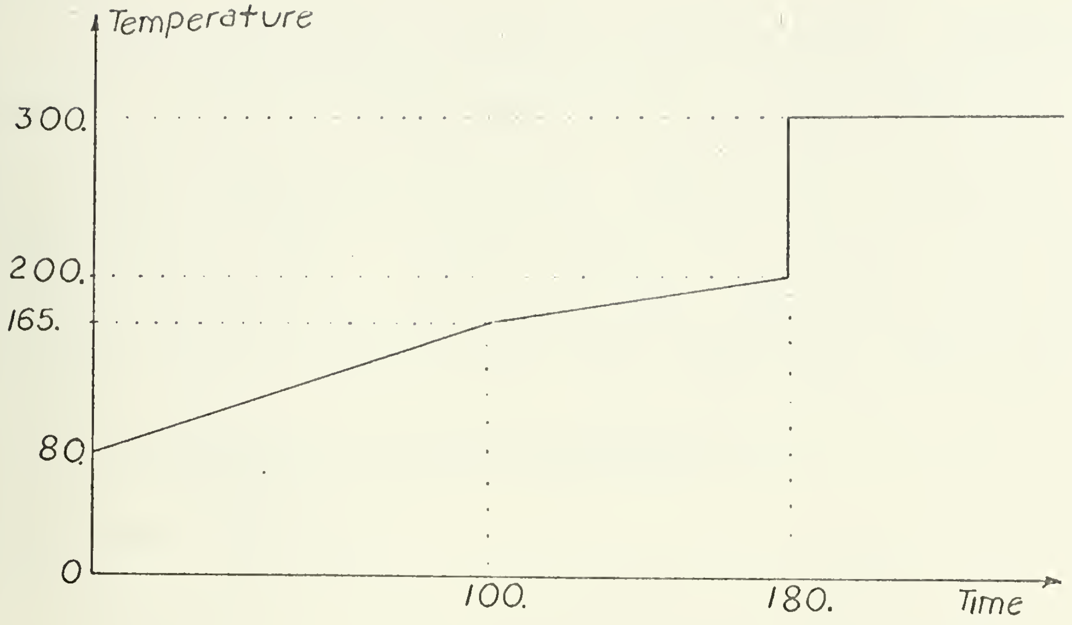


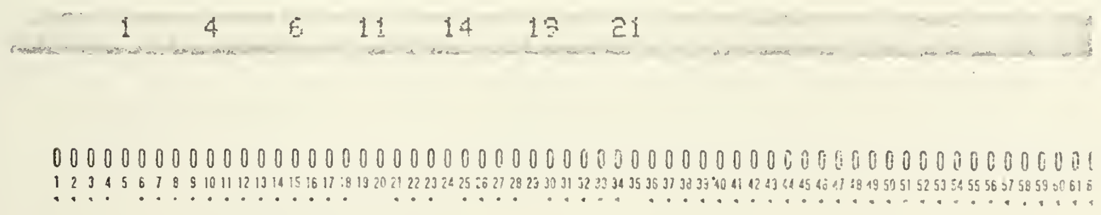
Figure 17. Entry Temperature-Time Variation.

Further details of this temperature-time history are specified in step 24.

Step 23:

Here we read in the global numbers of those nodes which are in contact with inside fluid, (NCFI(I)) with the format (12I5). It is important to read these node numbers in the direction of flow velocity.

For example, if the inside flow of Fig. 16 is from left to right, then for this step the card reads:



```
    1    4    6   11   14   19   21
```

0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61

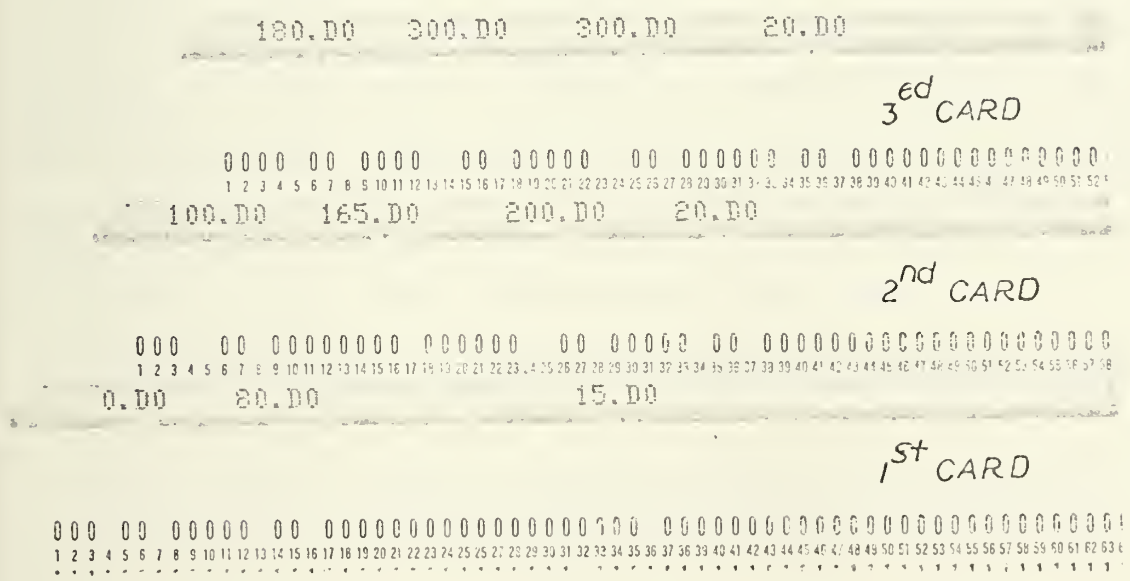
Step 24:

In this step data for inside flow are given. For each ramp I we read: the time when the ramp starts BDRYI(I,1), the initial temperature of ramp I BDRYI(I,2), the final temperature of ramp I (specify only if it differs from the initial temperature of the next ramp) BDRYI(I,3), and the velocity of the fluid for this ramp BDRYI(I,4). We read in:

((BDRYI(I,J),J=1,4),I=1,NRAMPI)

with format (4F10.4).

For example of Fig. 17, let the flow velocity for the first ramp be 15 and that for the second and third segment be 20. Three cards are needed:



```
    180.D0    300.D0    300.D0    20.D0
```
$3^{ed}$ CARD

```
100.D0    165.D0    200.D0    20.D0
```
$2^{nd}$ CARD

```
0.D0      20.D0               15.D0
```
$1^{st}$ CARD

Step 25:

In this step we must specify the following items:

| | |
|---|---|
| DTI | The time increment (step size) of trapezoidal integration |
| TIME1 | The time when the first calculated temperature vector is to be stored for thermal stress evaluation |

| | |
|---|---|
| EVERY | The interval of time between the successive temperature vectors which are to be stored |
| IVEC | Total number of temperature vectors to be stored for thermal stress evaluation |
| INTP | The indication for printing the calculated temperature vectors |

If INTP = 0 there will be no temperature prints

If INTP = 1 the program will print the temperature after every step of time integration

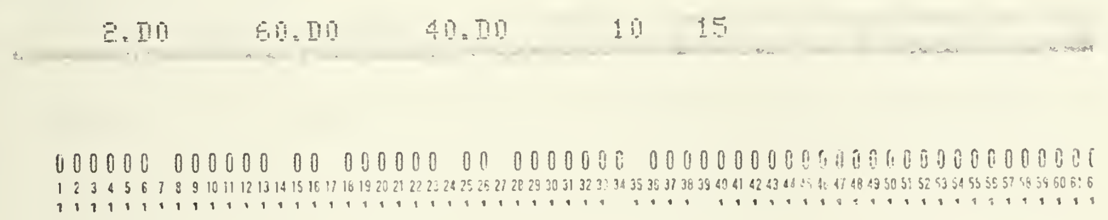
If INTP = 5 the program will print the temperature after every 5 steps of time integrations, etc.

At this step we prepare a single card that reads:

DTI, TIME1, EVERY, IVEC, INTP

with format (3F10.4,2I5).

For example, if the step size of trapezoidal time integration is 2 sec. and we desire to evaluate thermal stresses for 10 different temperature vectors, each 40 seconds apart, and starting with the first thermal stress calculation at time equal to 60, then the card for this step is:



```
    2.D0      60.D0     40.D0        10   15
```

where 15 in columns 34 and 35 indicates that the temperatures will be printed every 15 steps of time integration. The total time of integration for this example is 420 seconds, since

TIME1 + EVERY * (IVEC-1) = 420.

Step 26:

If there are no known temperature vectors (IEXT = 0) for which thermal stress evaluation is desired, omit this step.

For IEXT > 0 we must specify the nodal temperatures which are to be stored for thermal stress evaluation. We read:

```
((STØR (I,J) , J=1,NNT), I=1,IEXT)
```

with the format (6F10.4).

This means that we enter all the components of the first nodal temperature vector, followed by the components of the second nodal temperature vector and so on until IEXT such vectors have been entered.

Step 27:

The data cards for this problem are completed now. If no more problems are to be solved for the same geometry, omit this step.

If another problem is to be solved for the same geometry and spatial discretization, increment by 1 the NPRØB in step 4 and start the input of the new problem from step 10.

Step 28:

If there is no access to the IBM 360 computer of the Naval Postgraduate School, omit the following steps and start from step 32.

For the convenience of the user, the author has put a so-called CHECK program and the program presented in

Appendix B in the computer system at N.P.S. (Naval Post-graduate School). It is advised, however, that the user check his input data deck with the CHECK program before attempting to solve the problem.

For using the CHECK program prepare the following control cards.

```
//XXXX0000 JØB (0000,0000FT,XX00),'NAME',TIME=1
//JØBLIB  DD  DSN=F0609.BAKH,DISP=SHR,UNIT=2314,VØL=SER=DUFFY
//GØ  EXEC  PGM=CHECK,REGION=100K
//FT06F001  DD SYSØUT=A,DCB=(RECFM=FBA,LRECL=133,BLKSIZE=3325),
//             UNIT=SYSØUT
//FT05F001  DD  *
```

where the first card is the regular FORTRAN job card used at this institution.

Prepare your deck as Fig. 18 and it may be read in from the hot card reader.

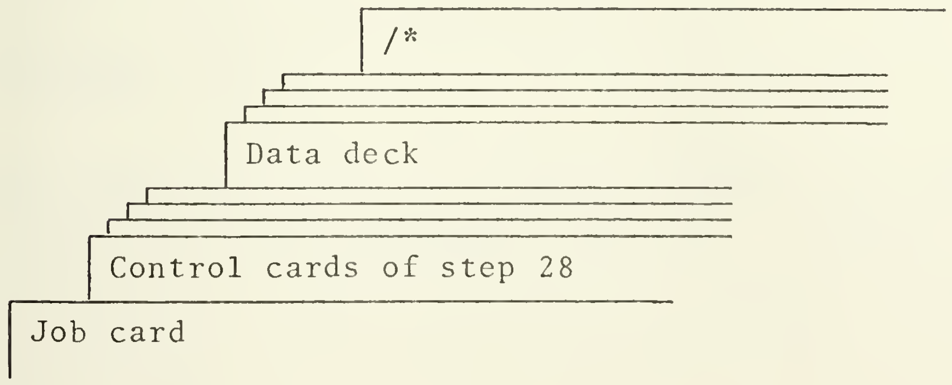


Figure 18. Set-up for Using CHECK Program.

If the output of the CHECK program has any error messages or has not been run, then there are some mistakes in the data deck or control cards.

If there are no error messages on the output of CHECK program, then study carefully the output and make sure it is the same problem that has been intended to be solved. Once the correctness of the input data has been insured, proceed to the next step.

Step 29:

AXITTS stands for Axisymmetric Transient Thermal Stress and this is the name given to the program presented in Appendix B when stored in the computer. For using that we must prepare the following control cards.

```
//XXXX0000 JØB (0000,000FT,XX00)'NAME',TIME=10
//JØBLIB  DD  DSN=F0609.BAKH,DISP=SHR,UNIT=2314,VØL=SER=DUFFY
//GØ  EXEC  PGM=AXITTS,REGION=425K
//FT06F001  DD SYSØUT=A,DCB=(RECFM=FBA,LRECL=133,BLKSIZE=3325),
//          UNIT=SYSØUT,SPACE=(CYL,(6,1))
//FT05F001  DD  *
```

where the first card is the regular FØRTRAN job card. It is advised to ask for 10 minutes time, i.e., TIME=10, since this would not affect the priority of the job within the class K jobs.

Prepare the deck as in Fig. 19 (it may be read into the computer from the hot card reader) and submit a so-called

service request card, available in the computer center, to the operator on duty.

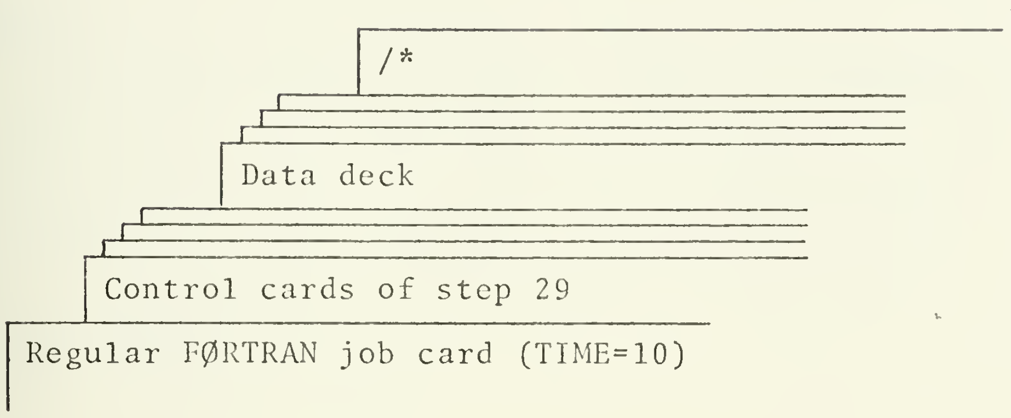


Figure 19. Deck Set-up for Using AXITTS Program.

<u>Step 30</u>:

If the user wishes to have a listing of the program he may prepare the following control cards as in Fig. 20 since the program is also listed on the data cell for this purpose.

```
Regular FØRTRAN job card
//PRNT  EXEC PGM=IEBPTCH
//SYSPRINT  DD DUMMY
//SYSUT1  DD DSN=F0609.BAKH,DISP=ØLD,UNIT=2321,
//    VØL=SER=CEL003,DCB=(RECFM=FB,LRECL=80,BLKSIZE=2000)
//SYSUT2  DD  SYSØUT=A,SPACE=(CYL,6)
//SYSIN  DD  *
       PRINT  TYPØRG=PØ,MAXFLDS=1,MAXNAME=1
       MEMBER  NAME=AXITTS
       RECØRD  FIELD=(80)
/*
```

Figure 20. Deck Set-up for Obtaining a Listing of the Program AXITTS.

The set-up deck of Fig. 20 may be read in from the hot card reader to get a listing of the program.

Step 31:

If the user wishes to obtain a deck of the program AXITTS he may prepare a deck as in Fig. 21 and read it in from the hot card reader. He must notify the operator on duty to be prepared for punching almost two boxes of IBM cards.

```
Regular FØRTRAN job card
//PUNCH  EXEC PGM=IEBPTCH
//SYSPRINT  DD DUMMY
//SYSUT1  DD DSN=F-0909.BAKH,DISP=ØLD,UNIT=2321,
//    VØL=SER=CEL003,DCB=(RECFM=FB,LRECL=80,BLKSIZE=2000)
//SYSUT2  DD  SYSØUT=B,SPACE=(CYL,6)
//SYSIN  DD  *
       PUNCH TYPØRG=PØ,MAXFLDS=1,MAXNAME=1
MEMBER  NAME=AXITTS
RECØRD  FIELD=(80)
/*
```

Figure 21. Deck Set-up for Obtaining a Punched Deck of the Program AXITTS

Step 32:

For using the program presented in Appendix B, if the user does not have access to the computer facility at N.P.S., he must punch a copy of the program. (Good luck!)

For the storage location 425K bytes are required and since the output may be longer than what usually is allowed, three additional cylinders are recommended. The execution time required depends on the size of the problem and the number of time integration steps; however, 10 minutes computer time would be sufficient for a problem of 140 nodes and 500 steps of time integration, with 20 temperature vectors stored for stress calculation.

# APPENDIX D

# PROGRAMMING

The computer program presented in Appendix B is formed from one main program and fifteen different subroutines. The communication between the main program and the subroutines is handled through the common blocks. In order to minimize storage requirements, most of the storage location of the system stiffness matrix is used for temperature calculations. This is accomplished by use of EQUIVALENCE statements. The total storage requirement is 425K bytes.

In this section the function of some parts of the program is discussed and the assumptions used are brought to attention.

## 1. MAIN PROGRAM

The main program is simply calling different subroutines when they are needed. The subroutines which are called in main program are:

INPUT, PRØB, CANDY, FLØW, FØRMV, TEMPER, PLØT, STIFF, FØRMF, CENTF, PRESS, DISPL, AND STRESS.

## 2. SUBROUTINE INPUT

In this subroutine the data regarding the geometry of the body, subdivisions into elements, and the material properties are read in and printed out. Calculation of the mid-side coordinates based on the straight line is done

here. This subroutine calls for subroutine PLØTRZ once to produce a graphical representation of the nodal points then the half-band-width of the system stiffness matrix is calculated from the connectivity array. Also, depending upon the choice of the system of units, the necessary conversions in each system of units are accomplished and the reference temperature (70°F or 20°C) based on the system of units is selected here.

## 3. SUBROUTINE PRØB

For each problem this subroutine is called once by the main program to read in and print out the information regarding the nature of the problem. Based upon the given information the type of the problem is distinguished here and for the thermal problems the length of time integration is calculated.

## 4. SUBROUTINE CANDY

Subroutine CANDY (C and Y) evaluates the capacitance matrix $\underline{\underline{C}}$ and admittance matrix $\underline{\underline{Y}}^+$ of Eq. 13, in banded form. For each problem this subroutine is called once. The functional flow chart of the subroutine CANDY is given in Fig. 22. Since the only non-zero elements of the symmetric matrix $\underline{\underline{Y}}^*$ (Eq. 11) are the diagonal, first and second super-diagonals, in order to conserve storage and reduce the number of arithmetic operations, the non-zero elements are stored in three different vectors (DIAG, ØFF1, ØFF2).

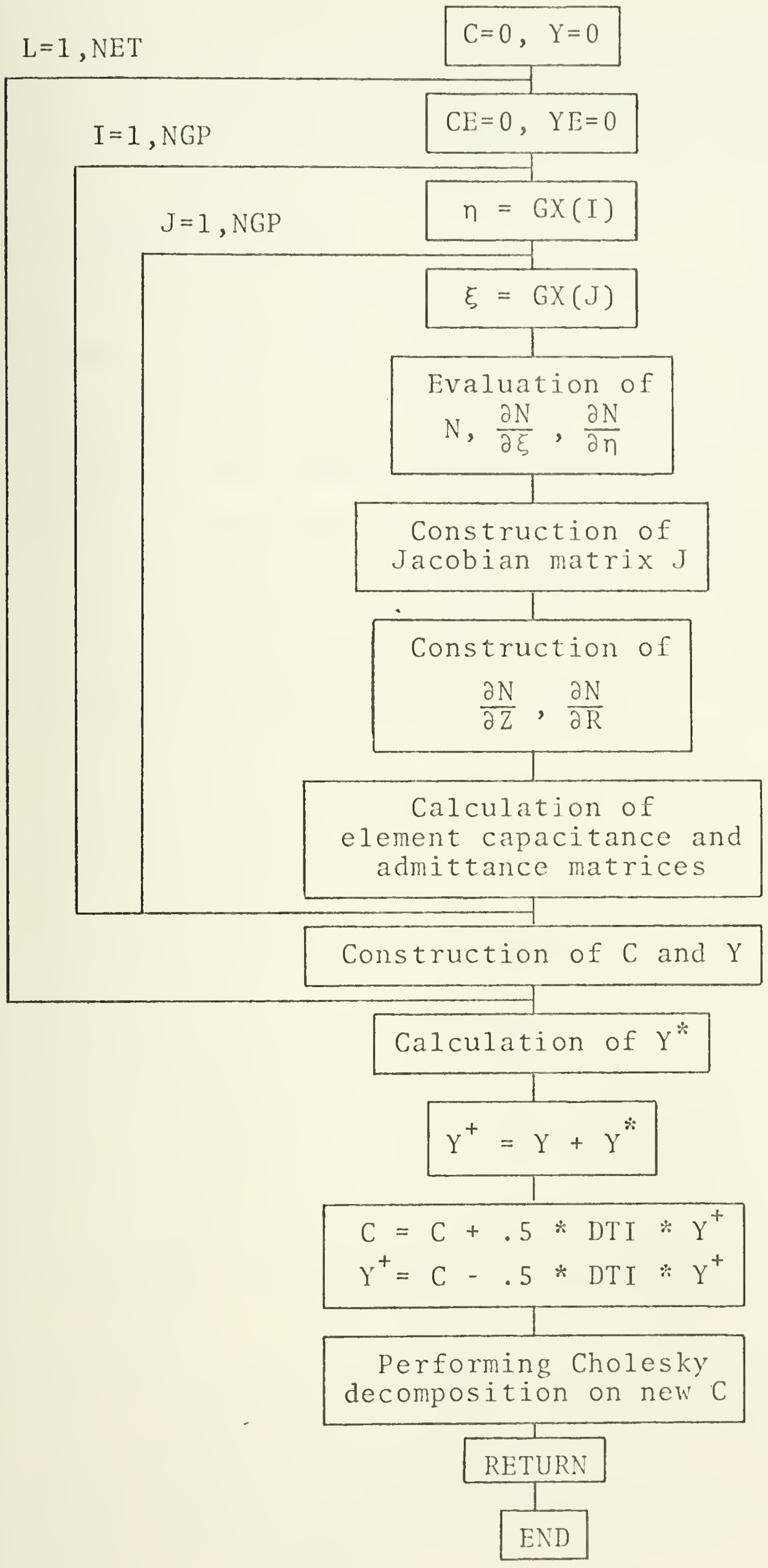

C=0, Y=0
L=1,NET
CE=0, YE=0
I=1,NGP
$\eta$ = GX(I)
J=1,NGP
$\xi$ = GX(J)
Evaluation of $N, \frac{\partial N}{\partial \xi}, \frac{\partial N}{\partial \eta}$
Construction of Jacobian matrix J
Construction of $\frac{\partial N}{\partial Z}, \frac{\partial N}{\partial R}$
Calculation of element capacitance and admittance matrices
Construction of C and Y
Calculation of $Y^*$
$Y^+ = Y + Y^*$
$C = C + .5 * DTI * Y^+$
$Y^+ = C - .5 * DTI * Y^+$
Performing Cholesky decomposition on new C
RETURN
END

Figure 22. Functional Flow Chart of CANDY.

Once the system capacitance and admittance matrices $\underline{\underline{C}}$ and $\underline{\underline{Y}}^{+}$ are formed, then the matrices $\underline{\underline{A}}$ and $\underline{\underline{G}}$ (Eq. 29) are evaluated and replace $\underline{\underline{C}}$ and $\underline{\underline{Y}}$ respectively.

## 5. SUBRØUTINE FLØW

If there is any convection or constant temperature thermal boundary condition, this subroutine is called from the main program for each step of time integration in order to evaluate the temperature of the fluid nodes. The calculation is based upon the constant fluid flux assumption (for both inside and outside flow). Consider a section of an irregular cylindrical pipe as Fig. 23 and focus attention on element i+1 on the inner boundary.

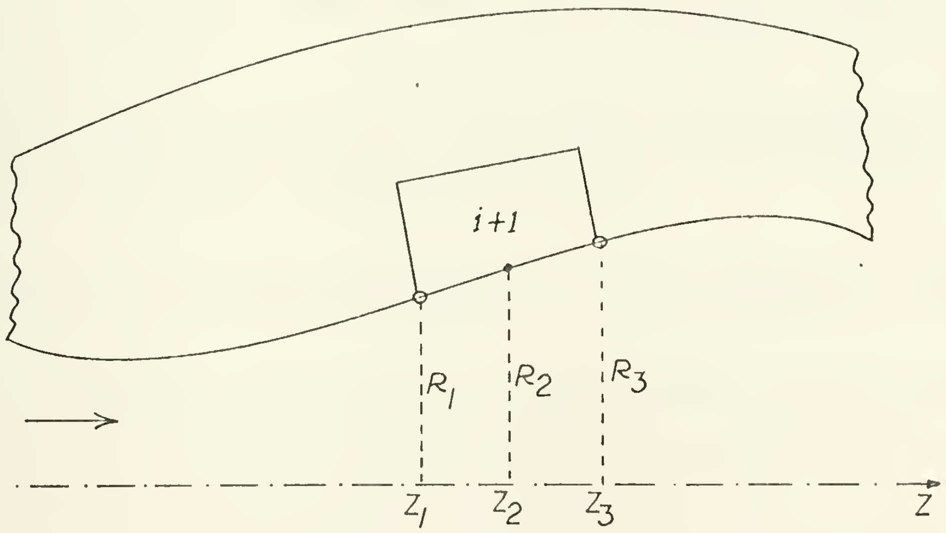


Figure 23. Fluid Node Representation for Inside Flow.

If fluid is moving from left to right we may number the R and Z coordinates of the fluid nodes of the element i+1 as in Fig. 23, then the volume of fluid pumped into the pipe at time $\tau$ is given by

$$Q_{in} = a\ v\ \tau, \tag{56}$$

where a and v are, respectively, the area and velocity at the entrance section. Also the volume of the pipe up to the mid-node of element i+1 is

$$Q_1 = V_i + \int_{Z_1}^{Z_2} \pi R^2\ dZ. \tag{57}$$

where $V_i$ is the volume of the pipe between the entrance section and element i+1 on the inner boundary.

Now we can write

$$R = \sum_{i=1}^{3} R_i\ N_i, \tag{58}$$

where $N_i$ are the one dimensional shape functions (given in Appendix A).

If we substitute Eq. 58 into 57 and make the coordinate transformation, after integration we get

$$Q_1 = V_i + \frac{\pi}{120}(Z_2 - Z_1)(31R_1^2 + 64R_2^2 + R_3^2 + 46R_1R_2 - 8R_1R_3 - 14R_2R_3). \tag{59}$$

Now by equating Eq. 59 to Eq. 56 at any time $\tau$ we may determine whether the front of the flow has already passed the mid-node of the wetted side of the element i+1.

A similar argument can be carried out for the end node of the element i+1. In that case we also get an equation similar to Eq. 59 with different numerical coefficients.

For the case of the outside flow it has also been assumed to have a constant fluid flux and a fictitious

entrance circular cross-sectional area (whose radius is the largest R plus unity) has been assumed.

The numerical coefficients of R's in Eq. 59 are used in subroutine FLØW and it has been assumed that the temperature of a fluid "particle" does not change during passage through the active section. This is believed to be an acceptable approximation for representative values of flow velocity and active length. With this assumption, for a given entry time-temperature relation (inside and outside flow) this subroutine evaluates the fluid nodal temperatures at any time.

## 6. SUBRØUTINE FØRMV

At every step of time integration this subroutine is called by the main program to form the vector $\underline{v}$ of the right-hand side of the discretized finite element (Eq. 13) for the given thermal boundary conditions. The program itself is self explanatory.

## 7. SUBRØUTINE TEMPER

The nodal temperatures are calculated in this subroutine with the trapezoidal time integrations. Irons' correction is applied here after every 10 steps of time integration. The temperature vectors selected for the stress analysis are stored. The transient temperatures will be printed out at the desired interval of time.

8. SUBRØUTINE STIFF

For any stress or thermal stress problem subroutine STIFF is called once by the main program to evaluate the stiffness matrix of the system. Once the stiffness matrix of the system is calculated then the desired structural boundary conditions are applied and, at the end, the Cholesky decomposition is performed. The functional flow chart of subroutine STIFF is given in Fig. 24.

9. FØRMF

In subroutine FØRMF the thermal load vectors are calculated for as many as (IVEC) given temperature vectors. These thermal load vectors are the columns of a rectangular matrix $\underline{\underline{F}}$. The provision is made that no matter how many temperature vectors are given (always IVEC $\leq$ 20) the IVEC number of columns of the $\underline{\underline{F}}$ are filled with the thermal load vectors and the last column of F will contain the load vector corresponding to the unit end displacement for zero axial force when the plane-end boundary condition is applied. The flow chart of subroutine FØRMF is given in Fig. 25.

10. SUBRØUTINE CENTF

If the system is rotating about the axis of revolution, this subroutine is called once by the main program to evaluate the centrifugal load vector. This load vector is always placed in the first column after the thermal load vectors (IVEC+1 position) in $\underline{\underline{F}}$.

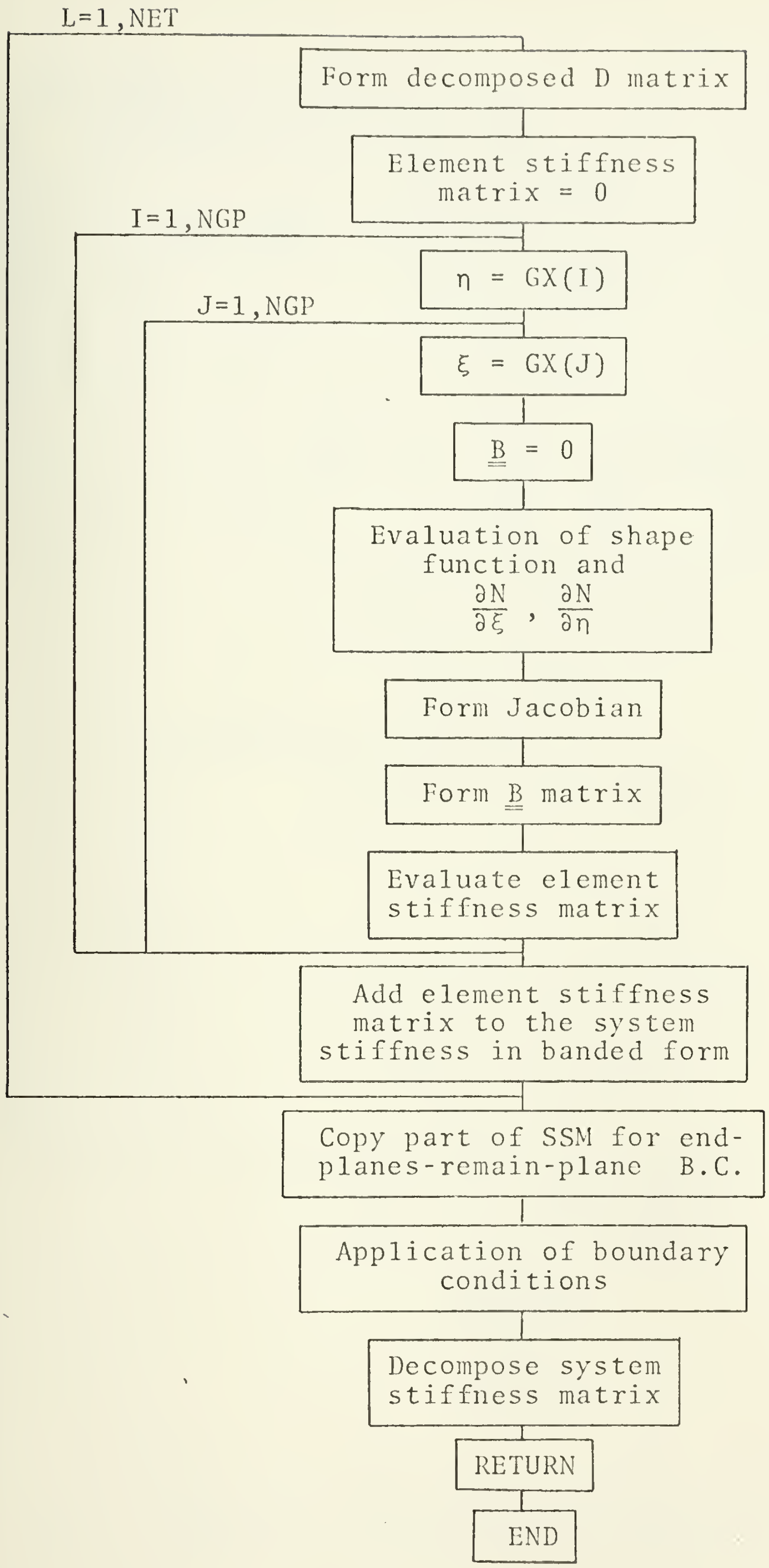

L=1,NET
Form decomposed D matrix
Element stiffness matrix = 0
I=1,NGP
η = GX(I)
J=1,NGP
ξ = GX(J)
B = 0
Evaluation of shape function and ∂N/∂ξ, ∂N/∂η
Form Jacobian
Form B matrix
Evaluate element stiffness matrix
Add element stiffness matrix to the system stiffness in banded form
Copy part of SSM for end-planes-remain-plane B.C.
Application of boundary conditions
Decompose system stiffness matrix
RETURN
END

Figure 24. Functional Flow Chart of STIFF.

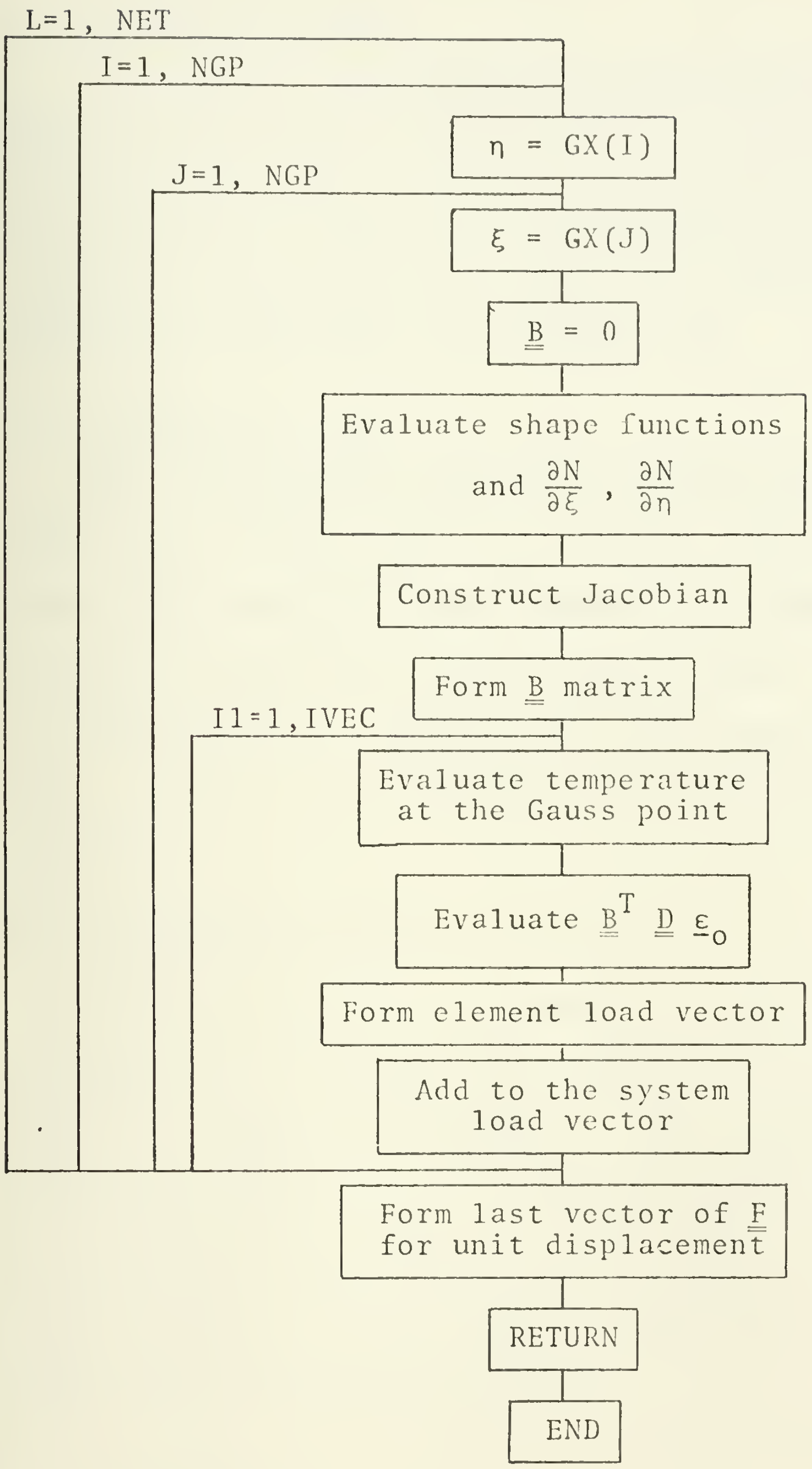

L=1, NET
I=1, NGP
$\eta$ = GX(I)
J=1, NGP
$\xi$ = GX(J)
$\underline{\underline{B}} = 0$
Evaluate shape functions and $\frac{\partial N}{\partial \xi}$ , $\frac{\partial N}{\partial \eta}$
Construct Jacobian
Form $\underline{\underline{B}}$ matrix
I1=1,IVEC
Evaluate temperature at the Gauss point
Evaluate $\underline{\underline{B}}^T \underline{\underline{D}} \underline{\varepsilon}_o$
Form element load vector
Add to the system load vector
Form last vector of $\underline{\underline{F}}$ for unit displacement
RETURN
END

Figure 25. Functional Flow Chart of FØRMF.

11. SUBRØUTINE PRESS

If there is any pressure acting on the system, the pressure load vector is calculated in this subroutine and this vector is placed in the column next to centrifugal load vector, if any, otherwise it will fill the position designated for centrifugal load vector in $\underline{\underline{F}}$.

12. SUBRØUTINE DISPL

The displacement vectors are found in this subroutine. Since the stiffness matrix is already decomposed in banded form, then the displacement vectors one after another are obtained by back and forward substitution. The principle of superposition is applied here and finally the axial force, if any, is corrected for the plane-end boundary condition.

13. SUBRØUTINE STRESS

For every problem this subroutine is called by the main program once to evaluate the stresses at the points $(\eta=\pm 1,\ \xi=\pm\frac{1}{\sqrt{3}})$ of each element. The transient stresses are calculated and printed for each displacement vector. For each problem the maximum mean stress and the maximum octahedral shearing stress, together with the corresponding times and locations, are printed.

14. LIMITATIONS

In the process of the development of the program discussed so far, it has been intended that all of the calculations and storage of data occur in the computer without

using any external devices such as disks or magnetic tapes so as to be able to solve any sizable problem in relatively short time.

For this reason the following limitations (Table VI) are set forth which give an overall size of 450K bytes to the program.

TABLE VI

MAXIMUM VALUES FOR PROGRAM PARAMETERS

| | | |
|---|---|---|
| NBAND | Half band-width of the system stiffness matrix | 66 |
| NMAT | Total number of different materials | 5 |
| NET | Total number of elements | 40 |
| NNT | Total number of nodes | 149 |
| NCFØT | Total number of nodes in contact with outside fluid | 37 |
| NCFIT | Total number of nodes in contact with inside fluid | 37 |
| NNRE | Total number of nodes of right-hand end | 37 |
| NNLT | Total number of nodes constrained against any longitudinal motion | 37 |
| NNRT | Total number of nodes constrained against any radial motion | 37 |
| IVEC | Total number of temperature vectors stored for thermal stresses | 20 |
| NRAMPØ | Total number of ramps for outside flow | 15 |
| NRAMPI | Total number of ramps for inside flow | 15 |
| NPNT | Total number of pressure nodes | 37 |

## LIST OF REFERENCES

1. Zienkiewicz, O. C., The Finite Element Method in Engineering Science. McGraw-Hill, 1971.

2. Arpaci, Vedat S., Conduction Heat Transfer, Addison-Wesley Publishing Company, 1966.

3. Collatz, Lothar, Numerical Treatment of Differential Equations. Third edition. Springer-Verlag, New York Inc., 1966.

4. Private communication from Bruce M. Irons to Robert E. Newton.

5. Chapman, J., Heat Transfer, Second edition. MacMillan Company, 1967.

6. Timoshenko, S. and Goodier, J. N., Theory of Elasticity, Second Edition, McGraw-Hill, 1951.

7. Fung, Y. C. Foundations of Solid Mechanics. Prentice-Hall, 1965.

8. Irons, B. M., International Journal for Numerical Methods in Engineering, Vol. 2, No. 1, p. 5-35, January-March 1970.

9. Boley, Bruno A. and Weiner, Jerome H., Theory of Thermal Stresses. John Wiley and Sons, Inc., 1962.

## INITIAL DISTRIBUTION LIST

| | | No. Copies |
|---|---|---|
| 1. | Defense Documentation Center<br>Cameron Station<br>Alexandria, Virginia 22314 | 2 |
| 2. | Library, Code 0212<br>Naval Postgraduate School<br>Monterey, California 93940 | 2 |
| 3. | Department of Mechanical Engineering<br>Naval Postgraduate School<br>Monterey, California 93940 | 2 |
| 4. | Imperial Iranian Naval Headquarters<br>Old Shemiran Road<br>Teheran, Iran | 1 |
| 5. | University of Teheran<br>Department of Mechanical Engineering<br>Shah Reza Avenue<br>Teheran, Iran | 1 |
| 6. | Prof. Robert E. Newton (Code 59Ne)<br>Department of Mechanical Engineering<br>Naval Postgraduate School<br>Monterey, California 93940 | 5 |
| 7. | Prof. John E. Brock (Code 59Bc)<br>Department of Mechanical Engineering<br>Naval Postgraduate School<br>Monterey, California 93940 | 1 |
| 8. | Prof. Paul Pucci (Code 59Pc)<br>Department Of Mechanical Engineering<br>Naval Postgraduate School<br>Monterey, California 93940 | 1 |
| 9. | Prof. Gilles Cantin (Code 59Ci)<br>Department of Mechanical Engineering<br>Naval Postgraduate School<br>Monterey, California 93940 | 1 |
| 10. | Lt. M. Bakhshandehpour<br>Zarrab Khaneh<br>Abhar St. No 44<br>Teheran, Iran | 2 |

DOCUMENT CONTROL DATA - R & D

*(Security classification of title, body of abstract and indexing annotation must be entered when the overall report is classified)*

RIGINATING ACTIVITY *(Corporate author)*

Naval Postgraduate School
Monterey, California 93940

2a. REPORT SECURITY CLASSIFICATION

Unclassified

2b. GROUP

EPORT TITLE

FINITE ELEMENT SOLUTION FOR AXISYMMETRIC TRANSIENT THERMAL STRESSES

ESCRIPTIVE NOTES *(Type of report and, inclusive dates)*

Mechanical Engineer's Thesis; June 1972

UTHOR(S) *(First name, middle initial, last name)*

Manouchehr Bakhshandehpour; Lieutenant, Imperial Iranian Navy

| EPORT DATE | 7a. TOTAL NO. OF PAGES | 7b. NO. OF REFS |
|---|---|---|
| June 1972 | 174 | 9 |

CONTRACT OR GRANT NO.

PROJECT NO.

9a. ORIGINATOR'S REPORT NUMBER(S)

9b. OTHER REPORT NO(S) *(Any other numbers that may be assigned this report)*

DISTRIBUTION STATEMENT

Approved for public release; distribution unlimited.

SUPPLEMENTARY NOTES

12. SPONSORING MILITARY ACTIVITY

Naval Postgraduate School
Monterey, California 93940

ABSTRACT

A finite element formulation for solving axisymmetric transient heat conduction and thermal stress problems is developed in this thesis. The governing equations of uncoupled, linear, isotropic thermoelasticity are discretized using quadratic isoparametric elements. A FORTRAN IV program, using double precision arithmetic, is presented. Compact storage techniques for banded symmetric matrices are used.

Comparisons between exact and computer solutions demonstrate close agreement for a number of test problems. Detailed instructions for using the program are included.

| KEY WORDS | LINK A | | LINK B | | LINK C | |
|---|---|---|---|---|---|---|
| | ROLE | WT | ROLE | WT | ROLE | WT |
| AXISYMMETRIC TRANSIENT HEAT CONDUCTION | | | | | | |
| FINITE ELEMENT HEAT CONDUCTION | | | | | | |
| FINITE ELEMENT TRANSIENT THERMAL STRESSES | | | | | | |
| AXISYMMETRIC TRANSIENT THERMAL STRESSES | | | | | | |

Thesis 134923
B1735 Bakhshandehpour
c.1 Finite element solution for axisymmetric transient thermal stresses.